

## Projeto de Avaliação

Paulo Miguel Maciel Silva

Nº 31506 – Regime Diurno

Professor

Luís Ferreira

Ano letivo 2024/2025

Licenciatura em Engenharia de Sistemas Informáticos

Escola Superior de Tecnologia

Instituto Politécnico do Cávado e do Ave

## ÍNDICE

Resumo .....	IV
1. Introdução.....	V
1.1. Motivação .....	V
1.2. Enquadramento .....	V
1.3. Objetivos .....	V
1.4. Metodologia do Trabalho .....	VI
1.5. Plano de Trabalho .....	VI
1.6. Estrutura do Documento .....	VI
2. Estado da Arte/Enquadramento Teórico .....	VII
2.1. Fundamentos Teóricos Listas .....	VII
2.1.1. Exemplos Práticos .....	VII
2.1.1.1. Lista Ligada: .....	VII
2.1.1.2. Inserir no Inicio: .....	VIII
2.1.1.3. Inserir no Meio: .....	VIII
2.1.1.4. Inserir no Fim: .....	VIII
2.1.1.5. Remover no Inicio: .....	IX
2.1.1.6. Remover no Meio: .....	IX
2.1.1.7. Remover no Fim: .....	IX
2.1.2. Efeito Nefasto Vertical:.....	X
2.1.3. Efeito Nefasto Horizontal:.....	X
2.1.4. Efeito Nefasto Diagonal:.....	X
2.2. Fundamentos Teóricos Grafos.....	XI
2.2.1. Exemplos Práticos .....	XI
2.2.1.1. Grafo simples:.....	XI
2.2.1.1.1. Busca em largura .....	XII
2.2.1.1.2. Busca em profundidade .....	XII

3.	Desenvolvimento.....	XIII
3.1.	Análise e Especificação.....	XIII
3.1.1.	Fase 1:.....	XIII
3.1.2.	Fase 2:.....	XIII
3.2.	Implementação.....	XIII
4.	Análise e Discussão de Resultados.....	XV
5.	Conclusão .....	XV
	Bibliografia.....	XV
	Glossário e Siglas.....	XV

## **ÍNDICE DE FIGURAS**

Figura 1 - Lista Ligada .....	VII
Figura 2 - Inserir no Início.....	VIII
Figura 3 - Inserir no Meio.....	VIII
Figura 4 - Inserir no Fim .....	VIII
Figura 5 - Remover no Inicio.....	IX
Figura 6 - Remover no Meio .....	IX
Figura 7 - Remover no fim .....	IX
Figura 8 - Efeito Nefasto Vertical .....	X
Figura 9 - Efeito Nefasto Horizontal .....	X
Figura 10 - Efeito Nefasto Diagonal .....	X
Figura 11 - Grafo Orientado.....	XI
Figura 12 - Busca em largura .....	XII
Figura 13 -Busca em profundidade.....	XII

## **Resumo**

O presente relatório descreve o desenvolvimento de um projeto realizado na unidade curricular de Estruturas de Dados Avançados, do 2.<sup>º</sup> semestre do 1.<sup>º</sup> ano da Licenciatura em Engenharia de Sistemas Informáticos.

Este projeto permitiu aplicar conhecimentos adquiridos ao longo do semestre, incluindo leitura de ficheiros, manipulação de listas, remoção e inserção ordenada de antenas, criação de grafos e formas de percorrer um grafo.

A abordagem adotada consistiu na utilização de estruturas de dados dinâmicas e manipulação de ficheiros, oferecendo uma contribuição para a aplicação de conceitos fundamentais.

# **1. Introdução**

O presente capítulo pretende contextualizar o trabalho realizado, apresentando a motivação para o desenvolvimento, o enquadramento na unidade curricular, os objetivos definidos, a metodologia adotada, o planto de trabalho e a estrutura do documento.

## **1.1. Motivação**

O problema abordado neste projeto consiste na gestão de uma cidade com diversas antenas, onde cada uma transmite uma frequência específica. Na primeira fase, o desafio passa por detetar efeitos de interferência que surgem quando antenas com a mesma frequência se encontram alinhadas de uma determinada forma. Na segunda fase com essas antenas criadas, criar um grafo com as que tem a mesma frequência, bem como diferentes formas de o percorrer.

## **1.2. Enquadramento**

Este trabalho foi realizado no âmbito da unidade curricular de Estruturas de Dados Avançadas, integrada no do 2.º semestre do 1.º ano da Licenciatura em Engenharia de Sistemas Informáticos, na Escola Superior de Tecnologia do Instituto Politécnico do Cávado e do Ave.

## **1.3. Objetivos**

- Utilizar estruturas de dados dinâmicas para armazenar e manipular dados
- Gerir ficheiros texto e binários
- Desenvolver um algoritmo para inserir, remover e listar
- Calcular efeito nefasto das antenas
- Implementar grafos através da frequência das antenas
- Desenvolver algoritmos para atravessar grafos
- Implementar documentação completo em todo o código

## **1.4. Metodologia do Trabalho**

O trabalho foi desenvolvido em linguagem C, utilizando o Visual Studio Code, com foco na implementação progressiva de funcionalidades e na utilização da ferramenta Doxygen para documentação automática do código. A abordagem adotada foi gradual, o que permitiu uma revisão continua e o aperfeiçoamento constante do código.

O projeto na primeira fase, foi dividido em duas fases: na primeira, os primeiros dias foram dedicados a construção base do trabalho, incluindo a leitura do ficheiro e a implementação das funcionalidades de inserir, remover e listagem das antenas presentes no ficheiro texto. Na segunda fase, concentrei-me na melhoria do código e no cálculo do efeito nefasto. Devido a complexidade dessa fase, foi necessário um maior esforço e dedicação para garantir a sua implementação.

Na segunda parte do projeto, consistiu com a continuação base da fase1, adicionado a parte de criar um grafo com as que tivessem a mesma frequência, e algumas formas de percorrer o grafo.

## **1.5. Plano de Trabalho**

Inicialmente, o projeto foi pensado passo a passo com alguns rascunhos em papel para facilitar a compreensão do trabalho proposto, desde a leitura do ficheiro texto até aos algoritmos de percorrer o grafo.

## **1.6. Estrutura do Documento**

O documento encontra-se dividido nos seguintes capítulos: Introdução, Estado da Arte, Trabalho Desenvolvido, Análise de Resultados, Conclusão, Referências e Glossário.

## 2. Estado da Arte/Enquadramento Teórico

O presente capítulo tem como objetivo apresentar os conceitos fundamentais e soluções existentes relacionados com a manipulação de estruturas de dados dinâmicas.

### 2.1. Fundamentos Teóricos Listas

“Uma lista ligada é uma estrutura de dados que representa uma sequência de valores do mesmo tipo. É uma das estruturas de dados mais simples e mais utilizadas. Um elemento de uma lista ligada é tipicamente constituído por dois campos, um contendo um valor do elemento dessa posição e outro contendo uma referência (ou ligação) para o próximo elemento da lista. Esta forma de organização é exemplificada na figura seguinte, que ilustra uma lista ligada representando uma sequência de quatro números inteiros. Uma lista ligada pode ser usada como a implementação de vários tipos abstratos de dados, tais como filas, pilhas ou dicionários.” (Ribeiro, 2012)

#### 2.1.1. Exemplos Práticos

##### 2.1.1.1. Lista Ligada:

A figura 1 é um exemplo de uma lista ligada, onde a coordenada 2,3 é o início e tem um ponteiro para a próxima 5,7 e sempre assim, até a última que tem um apontador para NULL.

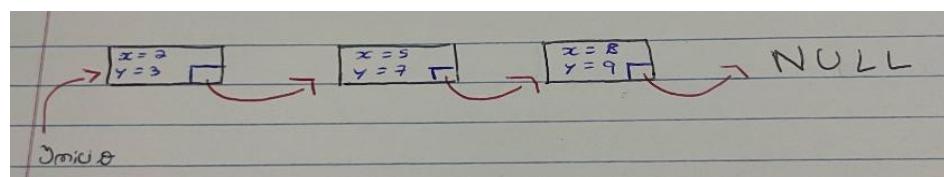


Figura 1 - Lista Ligada

### 2.1.1.2. Inserir no Início:

A figura 2, representa a inserção de uma nova antena no início da lista, onde a que estava em primeiro passa a ser apontada pela nova primeira.

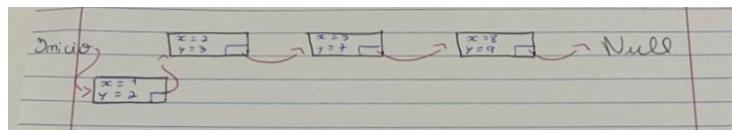


Figura 2 - Inserir no Início

### 2.1.1.3. Inserir no Meio:

A figura 3, representa a inserção de uma nova antena no meio da lista, por exemplo a antena 5,7 aponta para a nova e a nova vai apontar para a próxima.

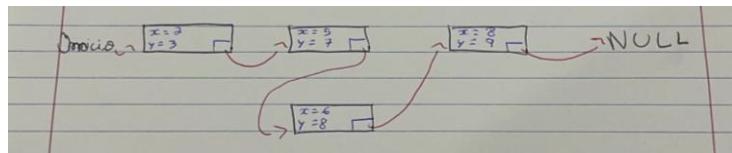


Figura 3 - Inserir no Meio

### 2.1.1.4. Inserir no Fim:

A figura 4, representa a inserção de uma nova antena no fim da lista, a antena nova vai ser apontada pela antiga última e vai apontar agora para o NULL.

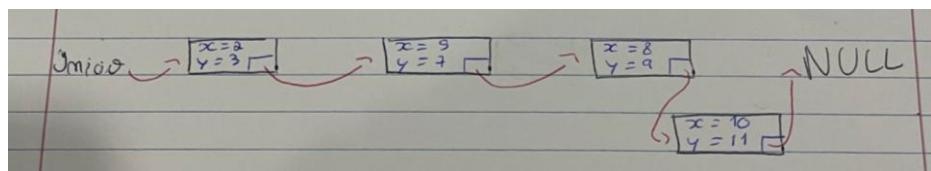


Figura 4 - Inserir no Fim

### 2.1.1.5. Remover no Inicio:

A figura 5, representa a remoção de uma antena no início da lista, a antena que estava em primeiro desaparece e a segunda para a ser a primeira, neste caso não tem de ser apontar para a próxima para não perder a lista toda.

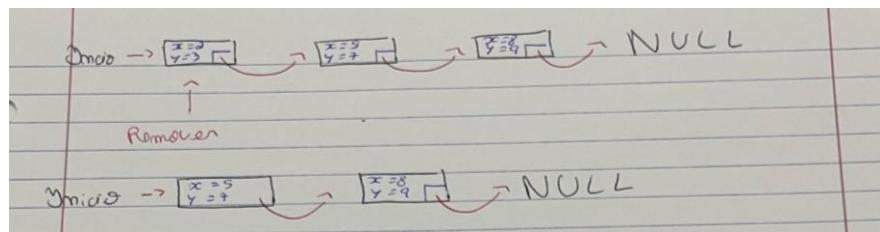


Figura 5 - Remover no Inicio

### 2.1.1.6. Remover no Meio:

A figura 6, representa a remoção de uma antena no meio da lista, na remoção da antena 5,7, a primeira vai apontar para a próxima a seguir da que foi eliminada.

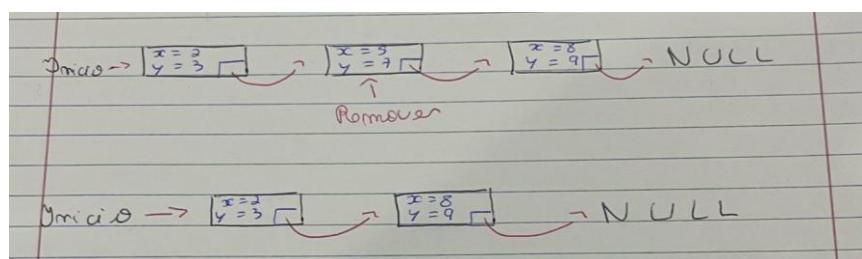


Figura 6 - Remover no Meio

### 2.1.1.7. Remover no Fim:

A figura 7, representa a remoção de uma antena no fim da lista, a anterior a que foi eliminada vai apontar para NULL.

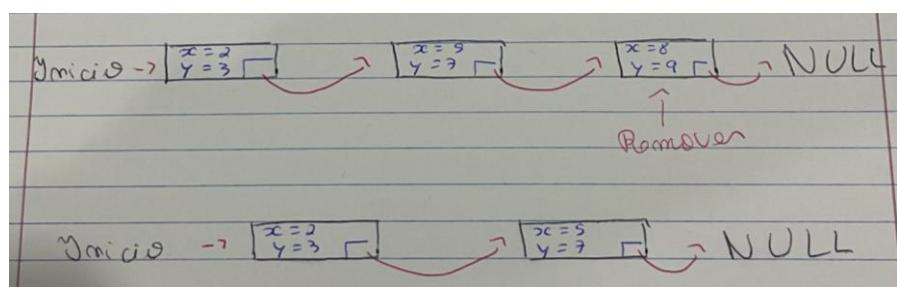


Figura 7 - Remover no fim

### 2.1.2. Efeito Nefasto Vertical:

A figura 8, representa o cálculo do efeito nefasto na vertical, se o y for igual ao y, tira 2 coordenadas ao y menor e soma mais 2 ao y maior.

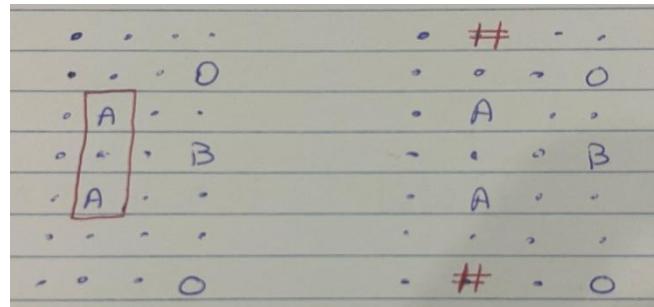


Figura 8 - Efeito Nefasto Vertical

### 2.1.3. Efeito Nefasto Horizontal:

A figura 9, representa o cálculo do efeito nefasto na horizontal, se o x for igual ao x, tira 2 coordenadas ao x menor e soma mais 2 ao x maior.

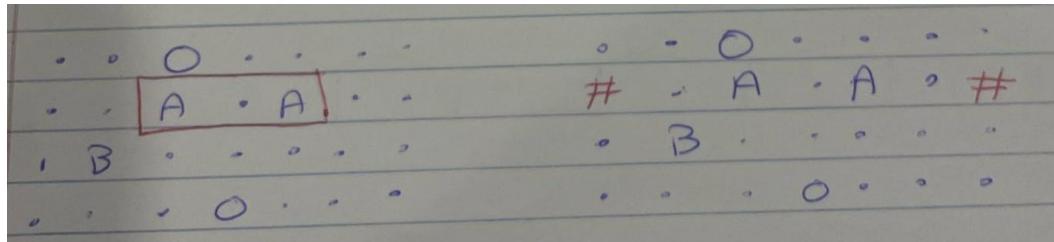


Figura 9 - Efeito Nefasto Horizontal

### 2.1.4. Efeito Nefasto Diagonal:

A figura 10, representa o cálculo do efeito nefasto na diagonal, onde se o  $x-2$  e  $y-2$  igual ao  $x+2$  e  $y+2$  então tira 2 no x e y menor e soma 2 no x e y maior.

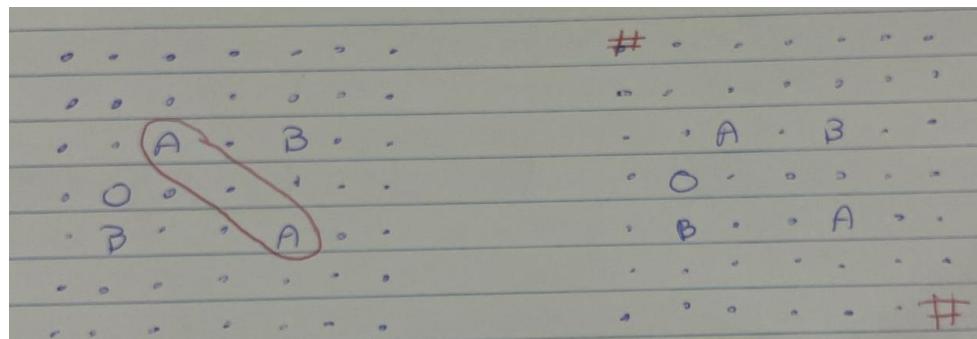
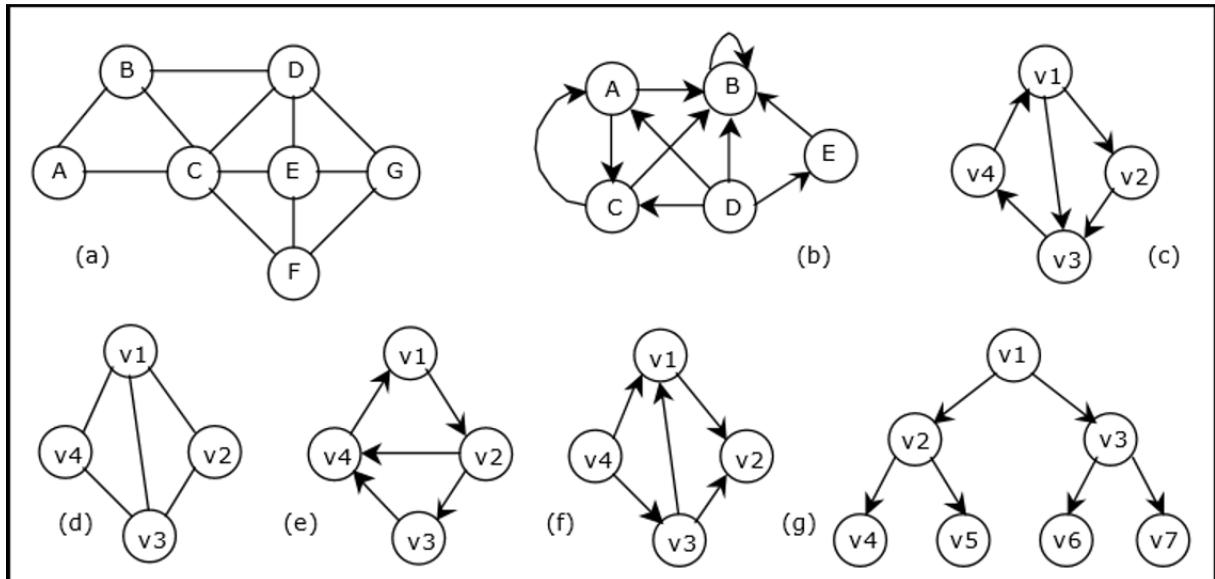


Figura 10 - Efeito Nefasto Diagonal

## 2.2. Fundamentos Teóricos Grafos

“Grafo é o conjunto dos vértices e as arestas que os ligam, pode ser orientado ou não orientado. “

$$G=(V,A)$$



” (Lufer, 2025)

### 2.2.1. Exemplos Práticos

#### 2.2.1.1. Grafo simples:

Exemplo simples de um grafo orientado no contexto do trabalho, neste caso as arestas são as que tem a mesma frequência.

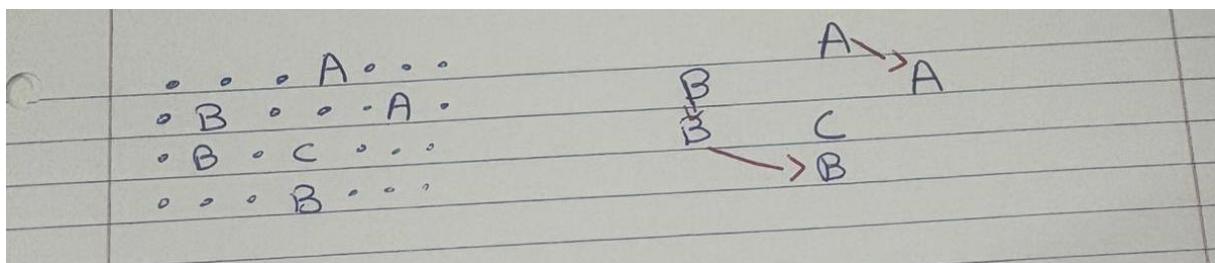


Figura 11 - Grafo Orientado

### 2.2.1.1. Busca em largura

Começa no primeiro vértice e visita todos aqueles vértices que estão ligados a ele, vai percorrendo o grafo camada a camada.

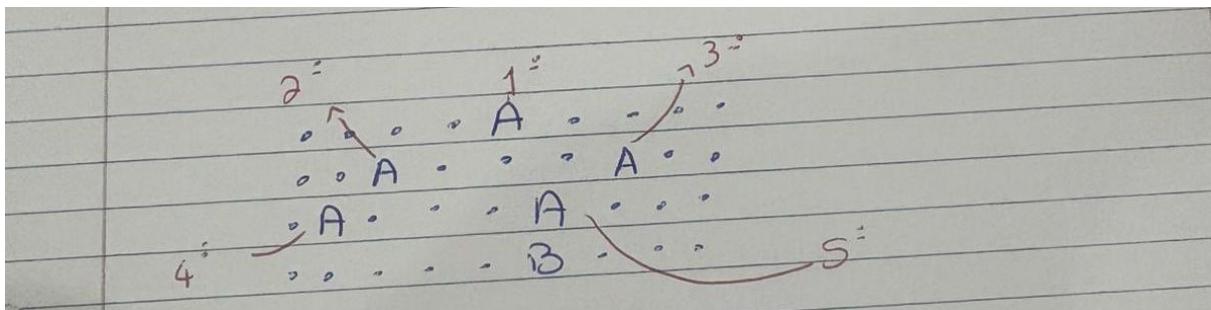


Figura 12 - Busca em largura

### 2.2.1.1. Busca em profundidade

Começa no primeiro vértice, seque para um vértice e vai ate ao ultimo vértice desse vértice, depois volta e faz o mesmo em todos, ate todos os vértices terem sido visitados

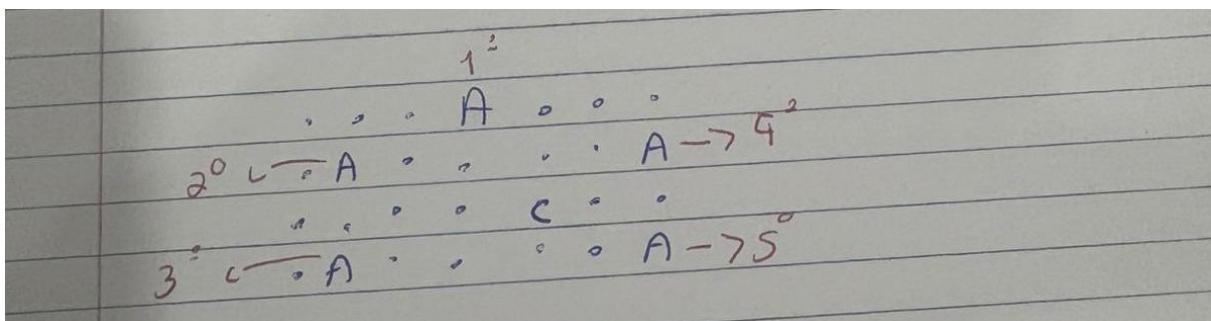


Figura 13 -Busca em profundidade

## **3. Desenvolvimento**

Este capítulo descreve a análise, especificação e implementação do projeto.

### **3.1. Análise e Especificação**

#### **3.1.1. Fase 1:**

- Requisitos funcionais: leitura de ficheiros, inserção ordenada, remoção, listagem e cálculo do efeito nefasto
- Estrutura: lista ligada simples com coordenadas
- Modelação: estruturas definidas em dados.h e as funções em funções.c

#### **3.1.2. Fase 2:**

- Requisitos funcionais: criação e travessias de grafos
- Estrutura: representação de grafos através da lista de adjacência
- Modelação: estruturas definidas em dados.h e as funções em funções.c numa pasta, e o main.c noutra.

### **3.2. Implementação**

A implementação decorreu em múltiplas fases:

- Criação da lista: Definição da struct Antena para representar cada antena com a sua frequência e coordenada.
- Leitura do ficheiro: Leitura linha a linha e conversão para os elementos da lista. Para tal exclui os ‘.’ e ‘\n’.
- Inserção ordenada: Inserção com validação de posição, assegurando a ordenação da lista por coordenadas ‘y’, em caso de ser igual, o ‘x’. Se a mesma já existir na mesma posição ou ao inserir for vazio não adiciona.
- Remoção: Função que remove as antenas por coordenadas, ajustando o ponteiro da lista.
- Listagem: Transformação da lista ligada num vetor para visualização.
- Cálculo Efeito Nefasto: Deteção das antenas com a mesma frequência alinhadas vertical, horizontal e na diagonal.
- Gravação: Criação do ficheiro antenas2.bin com recurso a uma estrutura auxiliar para guardar os dados.

- Ficheiro Main.c: Ficheiro responsável pela chamada e teste das funções.
- Documentação: Todo o código foi documentado utilizando a ferramenta Doxygen.
- Representação do grafo: Os grafos foram representados através de listas de adjacência, adequadas para gerir as ligações entre os nós.
- Implementação dos algoritmos de busca: Foram implementados os algoritmos de busca em largura (BFS) e busca em profundidade (DFS) para exploração eficiente do grafo.
- Objetivo: Esta implementação permitiu explorar as conexões dentro do grafo, identificando componentes e caminhos.

## 4. Análise e Discussão de Resultados

A trabalho desenvolvido cumpre os objetivos definidos. O programa permite gerir uma lista de antenas, assegurando as operações de inserir, remover e listar na fase 1, bem como a criação e formas de atravessar o grafo na fase2. A funcionalidade de cálculo o efeito nefasto foi implementado para casos verticais e horizontais e diagonais, com opção de escolher o tamanho do efeito. A validação das funcionalidades foi realizada com sucesso através de testes práticos.

## 5. Conclusão

Concluindo o projeto na Unidade Curricular de Estruturas de Dados Avançadas permitiu uma aplicação prática e aprofundada dos conhecimentos adquiridos ao longo do semestre.

A utilização de estruturas de dados dinâmicas, juntamente com a modularização adequada e a documentação detalhada, garantiu a clareza e a manutenção a longo prazo do código. Este projeto não só consolidou as competências técnicas adquiridas, mas também proporcionou uma melhor compreensão das boas práticas no desenvolvimento de software, nomeadamente no que diz respeito à eficiência e à organização do código.

Assim sendo todo o trabalho esta disponível no repositório [GitHub](#).

## Bibliografia

Lufer. (2025). Grafos. *Estrutura de Dados Avançadas*.

Ribeiro, P. (08 de 5 de 2012). *Lista Ligada*. Obtido de Casa Das Ciências: [https://wikiciencias.casadasciencias.org/wiki/index.php/Lista\\_Ligada](https://wikiciencias.casadasciencias.org/wiki/index.php/Lista_Ligada)

## Glossário e Siglas

- C – Linguagem de programação utilizada no desenvolvimento do projeto
- Doxygen – Ferramenta de documentação do código
- GitHub – Plataforma para alojamento e controlo do código
- Struct – Estrutura de dados em C para agrupar variáveis.