

Projeto - EDA Fase 2

Gerado por Doxygen 1.13.2

1 Índice das estruturas de dados	1
1.1 Estruturas de dados	1
2 Índice dos ficheiros	3
2.1 Lista de ficheiros	3
3 Documentação da estruturas de dados	5
3.1 Referência à estrutura Antena	5
3.1.1 Descrição detalhada	5
3.2 Referência à estrutura Aresta	5
3.2.1 Descrição detalhada	6
3.3 Referência à estrutura bfslista	6
3.3.1 Descrição detalhada	6
3.4 Referência à estrutura Caminho	6
3.4.1 Descrição detalhada	7
3.5 Referência à estrutura dfslista	7
3.5.1 Descrição detalhada	7
3.6 Referência à estrutura Fila	7
3.6.1 Descrição detalhada	7
3.7 Referência à estrutura Grafo	8
3.7.1 Descrição detalhada	8
3.8 Referência à estrutura interlista	8
3.8.1 Descrição detalhada	8
3.9 Referência à estrutura NoCaminho	9
3.9.1 Descrição detalhada	9
4 Documentação do ficheiro	11
4.1 estruturas.h	11
4.2 Referência ao ficheiro include/func.h	12
4.2.1 Descrição detalhada	13
4.2.2 Documentação das funções	13
4.2.2.1 adicionarAoCamino()	13
4.2.2.2 carregarAntenas()	14
4.2.2.3 criarAresta()	14
4.2.2.4 criarGrafo()	14
4.2.2.5 criarVertice()	15
4.2.2.6 desenfileirar()	15
4.2.2.7 detetarIntersecoes()	15
4.2.2.8 dfs()	15
4.2.2.9 dfsBacktracking()	16
4.2.2.10 encontrarCaminhoDFS()	16
4.2.2.11 enfileirar()	16
4.2.2.12 escreverAntenas()	17

4.2.2.13 escreverBFS()	17
4.2.2.14 escreverBloco()	17
4.2.2.15 escreverCaminho()	18
4.2.2.16 escreverDFS()	18
4.2.2.17 escreverIntersecoes()	18
4.2.2.18 execBfs()	19
4.2.2.19 execDfs()	19
4.2.2.20 filaVazia()	19
4.2.2.21 freeMem()	20
4.2.2.22 inicializarFila()	20
4.2.2.23 inserirArestas()	20
4.2.2.24 lerFicheiroEscreverOutput()	21
4.2.2.25 procurarAntenald()	21
4.2.2.26 removerDoCaminho()	21
4.2.2.27 segmentosSeCruzam()	22
4.3 func.h	22
4.4 Referência ao ficheiro src/func.c	23
4.4.1 Descrição detalhada	24
4.4.2 Documentação das funções	25
4.4.2.1 adicionarAoCamino()	25
4.4.2.2 carregarAntenas()	25
4.4.2.3 criarAresta()	25
4.4.2.4 criarGrafo()	26
4.4.2.5 criarVertice()	26
4.4.2.6 desenfileirar()	26
4.4.2.7 detetarIntersecoes()	26
4.4.2.8 dfs()	27
4.4.2.9 dfsBacktracking()	27
4.4.2.10 encontrarCaminhoDFS()	27
4.4.2.11 enfileirar()	28
4.4.2.12 escreverAntenas()	28
4.4.2.13 escreverBFS()	28
4.4.2.14 escreverBloco()	29
4.4.2.15 escreverCaminho()	29
4.4.2.16 escreverDFS()	29
4.4.2.17 escreverIntersecoes()	30
4.4.2.18 execBfs()	30
4.4.2.19 execDfs()	30
4.4.2.20 filaVazia()	31
4.4.2.21 freeMem()	31
4.4.2.22 inicializarFila()	31
4.4.2.23 inserirArestas()	32

4.4.2.24 lerFicheiroEscreverOutput()	32
4.4.2.25 procurarAntenaId()	32
4.4.2.26 removerDoCaminho()	33
4.4.2.27 segmentosSeCruzam()	33

Índice	35
---------------	-----------

Capítulo 1

Índice das estruturas de dados

1.1 Estruturas de dados

Lista das estruturas de dados com uma breve descrição:

Antena	Estrutura de uma antena (vértice)	5
Aresta	Estrutura de uma aresta	5
bfslista	Estrutura de uma lista de resultados da pesquisa em largura (bfs)	6
Caminho	Estrutura de um caminho (lista ligada)	6
dfslista	Estrutura de uma lista de resultados da pesquisa em profundidade (dfs)	7
Fila	Estrutura de uma Fila (lista dinâmica para bfs)	7
Grafo	Estrutura do Gráfo	8
interlista	Estrutura de uma lista de resultados de interseção	8
NoCaminho	Estrutura de um Nó de um caminho	9

Capítulo 2

Índice dos ficheiros

2.1 Lista de ficheiros

Lista de todos os ficheiros documentados com uma breve descrição:

include/estruturas.h	11
include/func.h	12
src/func.c	23

Capítulo 3

Documentação da estruturas de dados

3.1 Referência à estrutura Antena

Estrutura de uma antena (vértice)

```
#include <estruturas.h>
```

Campos de Dados

- int **x**
- int **y**
- char **freq**
- int **visitado**
- int **id**
- struct **Aresta** * **adj**
- struct **Antena** * **next**

3.1.1 Descrição detalhada

Estrutura de uma antena (vértice)

A documentação para esta estrutura foi gerada a partir do seguinte ficheiro:

- include/estruturas.h

3.2 Referência à estrutura Aresta

Estrutura de uma aresta.

```
#include <estruturas.h>
```

Campos de Dados

- struct [Antena](#) * destino
- struct [Aresta](#) * next

3.2.1 Descrição detalhada

Estrutura de uma aresta.

A documentação para esta estrutura foi gerada a partir do seguinte ficheiro:

- include/estruturas.h

3.3 Referência à estrutura bfslista

Estrutura de uma lista de resultados da pesquisa em largura (bfs)

```
#include <estruturas.h>
```

Campos de Dados

- [Antena](#) * antena
- struct [bfslista](#) * prox

3.3.1 Descrição detalhada

Estrutura de uma lista de resultados da pesquisa em largura (bfs)

A documentação para esta estrutura foi gerada a partir do seguinte ficheiro:

- include/estruturas.h

3.4 Referência à estrutura Caminho

Estrutura de um caminho (lista ligada)

```
#include <estruturas.h>
```

Campos de Dados

- int tamanho
- struct [NoCaminho](#) * head

3.4.1 Descrição detalhada

Estrutura de um caminho (lista ligada)

A documentação para esta estrutura foi gerada a partir do seguinte ficheiro:

- include/estruturas.h

3.5 Referência à estrutura dfslista

Estrutura de uma lista de resultados da pesquisa em profundidade (dfs)

```
#include <estruturas.h>
```

Campos de Dados

- Antena * antena
- struct dfslista * prox

3.5.1 Descrição detalhada

Estrutura de uma lista de resultados da pesquisa em profundidade (dfs)

A documentação para esta estrutura foi gerada a partir do seguinte ficheiro:

- include/estruturas.h

3.6 Referência à estrutura Fila

Estrutura de uma Fila (lista dinâmica para bfs)

```
#include <estruturas.h>
```

Campos de Dados

- bfslista * inicio
- bfslista * fim

3.6.1 Descrição detalhada

Estrutura de uma Fila (lista dinâmica para bfs)

A documentação para esta estrutura foi gerada a partir do seguinte ficheiro:

- include/estruturas.h

3.7 Referência à estrutura Grafo

Estrutura do Gráfo.

```
#include <estruturas.h>
```

Campos de Dados

- int **V**
- struct **Antena** * **vertices**

3.7.1 Descrição detalhada

Estrutura do Gráfo.

A documentação para esta estrutura foi gerada a partir do seguinte ficheiro:

- include/estruturas.h

3.8 Referência à estrutura interlista

Estrutura de uma lista de resultados de interseção.

```
#include <estruturas.h>
```

Campos de Dados

- int **x1**
- int **y1**
- int **x2**
- int **y2**
- int **x3**
- int **y3**
- int **x4**
- int **y4**
- char **freq**
- struct **interlista** * **prox**

3.8.1 Descrição detalhada

Estrutura de uma lista de resultados de interseção.

A documentação para esta estrutura foi gerada a partir do seguinte ficheiro:

- include/estruturas.h

3.9 Referência à estrutura NoCaminho

Estrutura de um Nó de um caminho.

```
#include <estruturas.h>
```

Campos de Dados

- int **id_antena**
- struct **NoCaminho** * **prox**

3.9.1 Descrição detalhada

Estrutura de um Nó de um caminho.

A documentação para esta estrutura foi gerada a partir do seguinte ficheiro:

- include/estruturas.h

Capítulo 4

Documentação do ficheiro

4.1 estruturas.h

```
00001
00009
00010 #ifndef ESTRUTURAS_H
00011 #define ESTRUTURAS_H
00012
00016
00017 typedef struct Antena
00018 {
00019     int x, y;           // coordenadas da antena
00020     char freq;          // frequência
00021     int visitado;       // inteiro para verificar se foi visitado
00022     int id;            // id
00023     struct Aresta *adj; // lista de adjacência (arestas)
00024     struct Antena *next; // pointer prox antena
00025 } Antena;
00026
00030
00031 typedef struct Aresta
00032 {
00033     struct Antena *destino; // proxima antena
00034     struct Aresta *next;    // pointer prox aresta
00035 } Aresta;
00036
00040
00041 typedef struct Grafo
00042 {
00043     int V;           // número de vértices
00044     struct Antena *vertices; // pointer para a lista ligada de antenas (head)
00045 } Grafo;
00046
00050
00051 typedef struct dfslista
00052 {
00053     Antena *antena; // antena
00054     struct dfslista *prox; // pointer prox antena
00055 } dfslista;
00056
00060
00061 typedef struct bfslista
00062 {
00063     Antena *antena; // antena
00064     struct bfslista *prox; // pointer prox antena
00065 } bfslista;
00066
00070
00071 typedef struct Fila
00072 {
00073     bfslista *inicio; // inicio da fila
00074     bfslista *fim;    // fim da fila
00075 } Fila;
00076
00080
00081 typedef struct Caminho
00082 {
00083     int tamanho; // número de saltos em um caminho
00084     struct NoCaminho *head; // inicio da lista
00085 } Caminho;
```

```

00086
00090
00091 typedef struct NoCaminho
00092 {
00093     int id_antena;           // id da antena no nó do caminho
00094     struct NoCaminho *prox; // pointer para o prox
00095 } NoCaminho;
00096
00100
00101 typedef struct interlista
00102 {
00103     int x1, y1, x2, y2;      // coordenadas do primeiro segmento
00104     int x3, y3, x4, y4;      // coordenadas do segundo segmento
00105     char freq;               // frequencia das antenas
00106     struct interlista *prox; // pointer para o prox
00107 } interlista;
00108
00109 #endif

```

4.2 Referência ao ficheiro include/func.h

```
#include "estruturas.h"
```

Funções

- **Grafo * criarGrafo ()**
Função para criar e inicializar um grafo.
- **Antena * criarVertice (Grafo *grafo, int x, int y, char freq)**
Função para criar uma antena/vértice no grafo.
- **int criarAresta (Antena *origem, Antena *destino)**
Função para criar uma aresta para ligar duas antenas.
- **Grafo * carregarAntenas (const char *ficheiro)**
Função principal de carregamento de antenas.
- **int inserirArestas (Grafo *grafo)**
Função para criar uma aresta de 2 antenas com a mesma função.
- **Antena * procurarAntenald (Grafo *grafo, int id)**
Função para procurar uma antena no grafo pelo id.
- **dfslista * dfs (Antena *atual, dfslista *lista)**
Função para fazer pesquisa em profundidade (DFS)
- **dfslista * execDfs (Grafo *grafo, int id_inicio)**
Função para executar o dfs através de um id de antena.
- **int inicializarFila (Fila *f)**
Função para inicializar a estrutura de uma fila vazia.
- **int enfileirar (Fila *f, Antena *a)**
Função que insere uma antena no final da fila.
- **Antena * desenfileirar (Fila *f)**
Função para remover a antena do inicio da lista.
- **int filaVazia (Fila *f)**
verifica se a fila está vazia
- **bfslista * execBfs (Grafo *grafo, int id_inicio)**
Função para fazer pesquisa em largura (bfs) com id de uma antena.
- **int adicionarAoCamino (Caminho *c, int id_antena)**
Função para adicionar uma nova antena (id) a um caminho.
- **int removerDoCaminho (Caminho *c)**
Função para remover uma antena (id) de um caminho.

- int `dfsBacktracking` (`Antena *atual`, `Antena *destino`, `Caminho *caminho`)
Função dfs para procurar caminhos.
- int `encontrarCaminhoDFS` (`Grafo *grafo`, int `id_origem`, int `id_destino`, `Caminho *caminho`)
Função principal para procurar caminhos entre 2 antenas no grafo.
- int `segmentosSeCruzam` (int `x1`, int `y1`, int `x2`, int `y2`, int `x3`, int `y3`, int `x4`, int `y4`)
Função para verificar se 2 segmentos se cruzam.
- `interlista *detetarIntersecoes` (`Grafo *grafo`)
Função para detetar interseções em um grafo.
- int `escreverBloco` (char `*tag`, FILE `*f`)
Função para escrever um bloco em um ficheiro bin.
- int `escreverAntenas` (`Grafo *grafo`, FILE `*f`)
Função para escrever antenas e os seus dados em um ficheiro bin.
- int `escreverDFS` (`dfslista *dfs`, FILE `*f`)
Função para escrever pontos encontrados pelo bfs em um ficheiro bin.
- int `escreverBFS` (`bfslista *bfs`, FILE `*f`)
Função para escrever pontos encontrados pelo bfs em um ficheiro bin.
- int `escreverCaminho` (`Caminho *caminho`, FILE `*f`)
Função para escrever caminhos em um ficheiro bin.
- int `escreverIntersecoes` (`interlista *lista`, FILE `*f`)
Função para escrever interseções em um ficheiro bin.
- int `lerFicheiroEscreverOutput` (const char `*nome_ficheiro`)
Função para ler e escrever na consola o conteúdo do ficheiro binário.
- int `freeMem` (`dfslista *dfs`, `bfslista *bfs`, `interlista *ints`)
Função para libertar memória.

4.2.1 Descrição detalhada

Autor

Bruno (a31496@alunos.ipca.pt)

Versão

1.2

Data

2025-05-18

Copyright

Copyright (c) 2025

4.2.2 Documentação das funções

4.2.2.1 adicionarAoCamino()

```
int adicionarAoCamino (
    Caminho * c,
    int id_antena)
```

Função para adicionar uma nova antena (id) a um caminho.

Parâmetros

<i>c</i>	caminho
<i>id_antena</i>	id da antena

Retorna

1 = sucesso

4.2.2.2 carregarAntenas()

```
Grafo * carregarAntenas (  
    const char * ficheiro)
```

Função principal de carregamento de antenas.

Parâmetros

<i>ficheiro</i>	nome do ficheiro txt com antenas
-----------------	----------------------------------

Retorna

grafo com antenas inseridas

4.2.2.3 criarAresta()

```
int criarAresta (  
    Antena * origem,  
    Antena * destino)
```

Função para criar uma aresta para ligar duas antenas.

Parâmetros

<i>origem</i>	antena de origem
<i>destino</i>	antena de destino

Retorna

1 = sucesso

4.2.2.4 criarGrafo()

```
Grafo * criarGrafo ()
```

Função para criar e inicializar um grafo.

Retorna

grafo criado

4.2.2.5 criarVertice()

```
Antena * criarVertice (  
    Grafo * grafo,  
    int x,  
    int y,  
    char freq)
```

Função para criar uma antena/vértice no grafo.

Parâmetros

<i>grafo</i>	grafo para inserir as antena
<i>x</i>	coordenada x da antena
<i>y</i>	coordenada y da antena
<i>freq</i>	frequência da antena

Retorna

pointer para a antena criada

4.2.2.6 desenfileirar()

```
Antena * desenfileirar (  
    Fila * f)
```

Função para remover a antena do inicio da lista.

Parâmetros

<i>f</i>	fila
----------	------

Retorna

antena do inicio da lista

4.2.2.7 detetarIntersecoes()

```
interlista * detetarIntersecoes (  
    Grafo * grafo)
```

Função para detetar interseções em um grafo.

Parâmetros

<i>grafo</i>	grafo com as antenas
--------------	----------------------

Retorna

Lista com os pontos que se interseitam

4.2.2.8 dfs()

```
dfslista * dfs (  
    Antena * atual,  
    dfslista * lista)
```

Função para fazer pesquisa em profundidade (DFS)

Parâmetros

<i>atual</i>	antena atual
<i>lista</i>	lista de antenas já visitadas

Retorna

lista com todas as antenas alcançadas pelo dfs

4.2.2.9 dfsBacktracking()

```
int dfsBacktracking (
    Antena * atual,
    Antena * destino,
    Caminho * caminho)
```

Função dfs para procurar caminhos.

Parâmetros

<i>atual</i>	antena atual
<i>destino</i>	antena destino
<i>caminho</i>	lista do caminho

Retorna

1 = caminho encontrado, 0 caso contrario

4.2.2.10 encontrarCaminhoDFS()

```
int encontrarCaminhoDFS (
    Grafo * grafo,
    int id_origem,
    int id_destino,
    Caminho * caminho)
```

Função principal para procurar caminhos entre 2 antenas no grafo.

Parâmetros

<i>grafo</i>	grafo com antenas
<i>id_origem</i>	id da antena de origem
<i>id_destino</i>	id da antena de origem
<i>caminho</i>	lista do caminho

Retorna

1 = caminho encontrado, 0 caso contrario

4.2.2.11 enfileirar()

```
int enfileirar (
    Fila * f,
    Antena * a)
```

Função que insere uma antena no final da fila.

Parâmetros

<i>f</i>	fila
<i>a</i>	antena que será inserida

Retorna

1 = sucesso

4.2.2.12 escreverAntenas()

```
int escreverAntenas (  
    Grafo * grafo,  
    FILE * f)
```

Função para escrever antenas e os seus dados em um ficheiro bin.

Parâmetros

<i>grafo</i>	grafo com antenas
<i>f</i>	ficheiro binário aberto

Retorna

1 = sucesso

4.2.2.13 escreverBFS()

```
int escreverBFS (  
    bfslista * bfs,  
    FILE * f)
```

Função para escrever pontos encontrados pelo bfs em um ficheiro bin.

Parâmetros

<i>bfs</i>	lista de pontos encontrados pelo bfs
<i>f</i>	ficheiro binário aberto

Retorna

1 = sucesso

4.2.2.14 escreverBloco()

```
int escreverBloco (  
    char * tag,  
    FILE * f)
```

Função para escrever um bloco em um ficheiro bin.

Parâmetros

<i>tag</i>	tag de 3 caracteres para identificar os dados exmpl "ANT" - antenas
<i>f</i>	ficheiro binário aberto

Retorna

1 = sucesso

4.2.2.15 escreverCaminho()

```
int escreverCaminho (  
    Caminho * caminho,  
    FILE * f)
```

Função para escrever caminhos em um ficheiro bin.

Parâmetros

<i>caminho</i>	lista de caminho
<i>f</i>	ficheiro binário aberto

Retorna

1 = sucesso

4.2.2.16 escreverDFS()

```
int escreverDFS (  
    dfslista * dfs,  
    FILE * f)
```

Função para escrever pontos encontrados pelo bfs em um ficheiro bin.

Parâmetros

<i>dfs</i>	lista de pontos encontrados pelo dfs
<i>f</i>	ficheiro binário aberto

Retorna

1 = sucesso

4.2.2.17 escreverIntersecoes()

```
int escreverIntersecoes (  
    interlista * lista,  
    FILE * f)
```

Função para escrever interseções em um ficheiro bin.

Parâmetros

<i>lista</i>	lista de inter
<i>f</i>	ficheiro binário aberto

Retorna

1 = sucesso

4.2.2.18 execBfs()

```
bfslista * execBfs (  
    Grafo * grafo,  
    int id_inicio)
```

Função para fazer pesquisa em largura (bfs) com id de uma antena.

Parâmetros

<i>grafo</i>	grafo com as antenas
<i>id_inicio</i>	id da antena inicial

Retorna

lista com as antenas visitadas pelo bfs

4.2.2.19 execDfs()

```
dfslista * execDfs (  
    Grafo * grafo,  
    int id_inicio)
```

Função para executar o dfs através de um id de antena.

Parâmetros

<i>grafo</i>	grafo com antenas
<i>id_inicio</i>	id da antena que começa a pesquisa

Retorna

lista com todos os pontos alcançados pelo dfs

4.2.2.20 filaVazia()

```
int filaVazia (  
    Fila * f)
```

verifica se a fila está vazia

Parâmetros

<i>f</i>	fila
----------	------

Retorna

1 se está vazia, caso contrario 0

4.2.2.21 freeMem()

```
int freeMem (  
    dfslista * dfs,  
    bfslista * bfs,  
    interlista * ints)
```

Função para libertar memória.

Parâmetros

<i>dfs</i>	lista de dfs
<i>bfs</i>	lista de dfs
<i>ints</i>	lista de interseções

Retorna

1 = sucesso

4.2.2.22 inicializarFila()

```
int inicializarFila (  
    Fila * f)
```

Função para inicializar a estrutura de uma fila vazia.

Parâmetros

<i>f</i>	fila que será inicializada
----------	----------------------------

Retorna

1 = sucesso

4.2.2.23 inserirArestas()

```
int inserirArestas (  
    Grafo * grafo)
```

Função para criar uma aresta de 2 antenas com a mesma função.

Parâmetros

<i>grafo</i>	grafo com antenas inseridas
--------------	-----------------------------

Retorna

1 = sucesso

4.2.2.24 lerFicheiroEscreverOutput()

```
int lerFicheiroEscreverOutput (  
    const char * nome_ficheiro)
```

Função para ler e escrever na consola o conteúdo do ficheiro binário.

Parâmetros

<i>nome_ficheiro</i>	nome do ficheiro binário
----------------------	--------------------------

Retorna

1 = sucesso

4.2.2.25 procurarAntenaId()

```
Antena * procurarAntenaId (  
    Grafo * grafo,  
    int id)
```

Função para procurar uma antena no grafo pelo id.

Parâmetros

<i>grafo</i>	grafo com as antenas
<i>id</i>	id da antena

Retorna

antena pedida

4.2.2.26 removerDoCaminho()

```
int removerDoCaminho (  
    Caminho * c)
```

Função para remover uma antena (id) de um caminho.

Parâmetros

c	caminho
---	---------

Retorna

1 = sucesso

4.2.2.27 segmentosSeCruzam()

```
int segmentosSeCruzam (  
    int x1,  
    int y1,  
    int x2,  
    int y2,  
    int x3,  
    int y3,  
    int x4,  
    int y4)
```

Função para verificar se 2 segmentos se cruzam.

Parâmetros

x1	coordenada x do primeiro ponto
y1	coordenada y do primeiro ponto
x2	coordenada x do segundo ponto
y2	coordenada y do segundo ponto
x3	coordenada x do terceiro ponto
y3	coordenada y do terceiro ponto
x4	coordenada x do quarto ponto
y4	coordenada y do quarto ponto

Retorna

1 se os segmentos (x1,y1)-(x2,y2) e (x3,y3)-(x4,y4) se cruzam, 0 caso contrário

4.3 func.h

[Ir para a documentação deste ficheiro.](#)

```
00001  
00009  
00010 #ifndef FUNC_H  
00011 #define FUNC_H  
00012  
00013 #include "estruturas.h"  
00014  
00015 Grafo *criarGrafo();  
00016 Antena *criarVertice(Grafo *grafo, int x, int y, char freq);  
00017 int criarAresta(Antena *origem, Antena *destino);  
00018  
00019 Grafo *carregarAntenas(const char *ficheiro);  
00020 int inserirArestas(Grafo *grafo);  
00021
```

```

00022 Antena *procurarAntenaId(Grafo *grafo, int id);
00023
00024 dfslista *dfs(Antena *atual, dfslista *lista);
00025 dfslista *execDfs(Grafo *grafo, int id_inicio);
00026
00027 int inicializarFila(Fila *f);
00028 int enfileirar(Fila *f, Antena *a);
00029 Antena *desenfileirar(Fila *f);
00030 int filaVazia(Fila *f);
00031 bfslista *execBfs(Grafo *grafo, int id_inicio);
00032
00033 int adicionarAoCamino(Caminho *c, int id_antena);
00034 int removerDoCamino(Caminho *c);
00035 int dfsBacktracking(Antena *atual, Antena *destino, Caminho *caminho);
00036 int encontrarCaminhoDFS(Grafo *grafo, int id_origem, int id_destino, Caminho *caminho);
00037
00038 int segmentosSeCruzam(int x1, int y1, int x2, int y2, int x3, int y3, int x4, int y4);
00039 interlista *detetarIntersecoes(Grafo *grafo);
00040
00041 int escreverBloco(char *tag, FILE *f);
00042 int escreverAntenas(Grafo *grafo, FILE *f);
00043 int escreverDFS(dfslista *dfs, FILE *f);
00044 int escreverBFS(bfslista *bfs, FILE *f);
00045 int escreverCaminho(Caminho *caminho, FILE *f);
00046 int escreverIntersecoes(interlista *lista, FILE *f);
00047 int lerFicheiroEscreverOutput(const char *nome_ficheiro);
00048
00049 int freeMem(dfslista *dfs, bfslista *bfs, interlista *ints);
00050
00051 #endif

```

4.4 Referência ao ficheiro src/func.c

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include "../include/estruturas.h"
#include "../include/func.h"

```

Funções

- **Grafo** * **criarGrafo** ()
Função para criar e inicializar um grafo.
- **Antena** * **criarVertice** (**Grafo** *grafo, int x, int y, char freq)
Função para criar uma antena/vértice no grafo.
- int **criarAresta** (**Antena** *origem, **Antena** *destino)
Função para criar uma aresta para ligar duas antenas.
- **Grafo** * **carregarAntenas** (const char *ficheiro)
Função principal de carregamento de antenas.
- int **inserirArestas** (**Grafo** *grafo)
Função para criar uma aresta de 2 antenas com a mesma função.
- **Antena** * **procurarAntenald** (**Grafo** *grafo, int id)
Função para procurar uma antena no grafo pelo id.
- **dfslista** * **dfs** (**Antena** *atual, **dfslista** *lista)
Função para fazer pesquisa em profundidade (DFS)
- **dfslista** * **execDfs** (**Grafo** *grafo, int id_inicio)
Função para executar o dfs através de um id de antena.
- int **inicializarFila** (**Fila** *f)
Função para inicializar a estrutura de uma fila vazia.

- int **enfileirar** (**Fila** *f, **Antena** *a)
Função que insere uma antena no final da fila.
- **Antena** * **desenfileirar** (**Fila** *f)
Função para remover a antena do início da lista.
- int **filaVazia** (**Fila** *f)
verifica se a fila está vazia
- **bfslista** * **execBfs** (**Grafo** *grafo, int id_inicio)
Função para fazer pesquisa em largura (bfs) com id de uma antena.
- int **adicionarAoCamino** (**Caminho** *c, int id_antena)
Função para adicionar uma nova antena (id) a um caminho.
- int **removerDoCaminho** (**Caminho** *c)
Função para remover uma antena (id) de um caminho.
- int **dfsBacktracking** (**Antena** *atual, **Antena** *destino, **Caminho** *caminho)
Função dfs para procurar caminhos.
- int **encontrarCaminhoDFS** (**Grafo** *grafo, int id_origem, int id_destino, **Caminho** *caminho)
Função principal para procurar caminhos entre 2 antenas no grafo.
- int **segmentosSeCruzam** (int x1, int y1, int x2, int y2, int x3, int y3, int x4, int y4)
Função para verificar se 2 segmentos se cruzam.
- **interlista** * **detetarIntersecoes** (**Grafo** *grafo)
Função para detetar interseções em um grafo.
- int **escreverBloco** (char *tag, FILE *f)
Função para escrever um bloco em um ficheiro bin.
- int **escreverAntenas** (**Grafo** *grafo, FILE *f)
Função para escrever antenas e os seus dados em um ficheiro bin.
- int **escreverDFS** (**bfslista** *dfs, FILE *f)
Função para escrever pontos encontrados pelo bfs em um ficheiro bin.
- int **escreverBFS** (**bfslista** *bfs, FILE *f)
Função para escrever pontos encontrados pelo bfs em um ficheiro bin.
- int **escreverCaminho** (**Caminho** *caminho, FILE *f)
Função para escrever caminhos em um ficheiro bin.
- int **escreverIntersecoes** (**interlista** *lista, FILE *f)
Função para escrever intersecoes em um ficheiro bin.
- int **lerFicheiroEscreverOutput** (const char *nome_ficheiro)
Função para ler e escrever na consola o conteudo do ficheiro binário.
- int **freeMem** (**bfslista** *dfs, **bfslista** *bfs, **interlista** *ints)
Função para libertar memória.

4.4.1 Descrição detalhada

Autor

Bruno (a31496@alunos.ipca.pt)

Versão

1.2

Data

2025-05-18

Copyright

Copyright (c) 2025

4.4.2 Documentação das funções

4.4.2.1 adicionarAoCamino()

```
int adicionarAoCamino (  
    Caminho * c,  
    int id_antena)
```

Função para adicionar uma nova antena (id) a um caminho.

Parâmetros

<i>c</i>	caminho
<i>id_antena</i>	id da antena

Retorna

1 = sucesso

4.4.2.2 carregarAntenas()

```
Grafo * carregarAntenas (  
    const char * ficheiro)
```

Função principal de carregamento de antenas.

Parâmetros

<i>ficheiro</i>	nome do ficheiro txt com antenas
-----------------	----------------------------------

Retorna

grafo com antenas inseridas

4.4.2.3 criarAresta()

```
int criarAresta (  
    Antena * origem,  
    Antena * destino)
```

Função para criar uma aresta para ligar duas antenas.

Parâmetros

<i>origem</i>	antena de origem
<i>destino</i>	antena de destino

Retorna

1 = sucesso

4.4.2.4 criarGrafo()

```
Grafo * criarGrafo ()
```

Função para criar e inicializar um grafo.

Retorna

grafo criado

4.4.2.5 criarVertice()

```
Antena * criarVertice (  
    Grafo * grafo,  
    int x,  
    int y,  
    char freq)
```

Função para criar uma antena/vértice no grafo.

Parâmetros

<i>grafo</i>	grafo para inserir as antena
<i>x</i>	coordenada x da antena
<i>y</i>	coordenada y da antena
<i>freq</i>	frequência da antena

Retorna

pointer para a antena criada

4.4.2.6 desenfileirar()

```
Antena * desenfileirar (  
    Fila * f)
```

Função para remover a antena do inicio da lista.

Parâmetros

<i>f</i>	fila
----------	------

Retorna

antena do inicio da lista

4.4.2.7 detetarIntersecoes()

```
interlista * detetarIntersecoes (  
    Grafo * grafo)
```

Função para detetar interseções em um grafo.

Parâmetros

<i>grafo</i>	grafo com as antenas
--------------	----------------------

Retorna

Lista com os pontos que se intersectam

4.4.2.8 dfs()

```
dfslista * dfs (  
    Antena * atual,  
    dfslista * lista)
```

Função para fazer pesquisa em profundidade (DFS)

Parâmetros

<i>atual</i>	antena atual
<i>lista</i>	lista de antenas já visitadas

Retorna

lista com todas as antenas alcançadas pelo dfs

4.4.2.9 dfsBacktracking()

```
int dfsBacktracking (  
    Antena * atual,  
    Antena * destino,  
    Caminho * caminho)
```

Função dfs para procurar caminhos.

Parâmetros

<i>atual</i>	antena atual
<i>destino</i>	antena destino
<i>caminho</i>	lista do caminho

Retorna

1 = caminho encontrado, 0 caso contrario

4.4.2.10 encontrarCaminhoDFS()

```
int encontrarCaminhoDFS (  
    Grafo * grafo,  
    int id_origem,  
    int id_destino,  
    Caminho * caminho)
```

Função principal para procurar caminhos entre 2 antenas no grafo.

Parâmetros

<i>grafo</i>	grafo com antenas
<i>id_origem</i>	id da antena de origem
<i>id_destino</i>	id da antena de origem
<i>caminho</i>	lista do caminho

Retorna

1 = caminho encontrado, 0 caso contrario

4.4.2.11 enfileirar()

```
int enfileirar (  
    Fila * f,  
    Antena * a)
```

Função que insere uma antena no final da fila.

Parâmetros

<i>f</i>	fila
<i>a</i>	antena que será inserida

Retorna

1 = sucesso

4.4.2.12 escreverAntenas()

```
int escreverAntenas (  
    Grafo * grafo,  
    FILE * f)
```

Função para escrever antenas e os seus dados em um ficheiro bin.

Parâmetros

<i>grafo</i>	grafo com antenas
<i>f</i>	ficheiro binário aberto

Retorna

1 = sucesso

4.4.2.13 escreverBFS()

```
int escreverBFS (  
    bfslista * bfs,  
    FILE * f)
```

Função para escrever pontos encontrados pelo bfs em um ficheiro bin.

Parâmetros

<i>bfs</i>	lista de pontos encontrados pelo bfs
<i>f</i>	ficheiro binário aberto

Retorna

1 = sucesso

4.4.2.14 escreverBloco()

```
int escreverBloco (  
    char * tag,  
    FILE * f)
```

Função para escrever um bloco em um ficheiro bin.

Parâmetros

<i>tag</i>	tag de 3 caracteres para identificar os dados exmpl "ANT" - antenas
<i>f</i>	ficheiro binário aberto

Retorna

1 = sucesso

4.4.2.15 escreverCaminho()

```
int escreverCaminho (  
    Caminho * caminho,  
    FILE * f)
```

Função para escrever caminhos em um ficheiro bin.

Parâmetros

<i>caminho</i>	lista de caminho
<i>f</i>	ficheiro binário aberto

Retorna

1 = sucesso

4.4.2.16 escreverDFS()

```
int escreverDFS (  
    dfslista * dfs,  
    FILE * f)
```

Função para escrever pontos encontrados pelo bfs em um ficheiro bin.

Parâmetros

<i>dfs</i>	lista de pontos encontrados pelo dfs
<i>f</i>	ficheiro binário aberto

Retorna

1 = sucesso

4.4.2.17 escreverIntersecoes()

```
int escreverIntersecoes (  
    interlista * lista,  
    FILE * f)
```

Função para escrever interseções em um ficheiro bin.

Parâmetros

<i>lista</i>	lista de inter
<i>f</i>	ficheiro binário aberto

Retorna

1 = sucesso

4.4.2.18 execBfs()

```
bfslista * execBfs (  
    Grafo * grafo,  
    int id_inicio)
```

Função para fazer pesquisa em largura (bfs) com id de uma antena.

Parâmetros

<i>grafo</i>	grafo com as antenas
<i>id_inicio</i>	id da antena inicial

Retorna

lista com as antenas visitadas pelo bfs

4.4.2.19 execDfs()

```
dfslista * execDfs (  
    Grafo * grafo,  
    int id_inicio)
```

Função para executar o dfs através de um id de antena.

Parâmetros

<i>grafo</i>	grafo com antenas
<i>id_inicio</i>	id da antena que começa a pesquisa

Retorna

lista com todos os pontos alcançados pelo dfs

4.4.2.20 filaVazia()

```
int filaVazia (  
    Fila * f)
```

verifica se a fila está vazia

Parâmetros

<i>f</i>	fila
----------	------

Retorna

1 se está vazia, caso contrario 0

4.4.2.21 freeMem()

```
int freeMem (  
    dfslista * dfs,  
    bfslista * bfs,  
    interlista * ints)
```

Função para libertar memória.

Parâmetros

<i>dfs</i>	lista de dfs
<i>bfs</i>	lista de dfs
<i>ints</i>	lista de interseções

Retorna

1 = sucesso

4.4.2.22 inicializarFila()

```
int inicializarFila (  
    Fila * f)
```

Função para inicializar a estrutura de uma fila vazia.

Parâmetros

<i>f</i>	fila que será inicializada
----------	----------------------------

Retorna

1 = sucesso

4.4.2.23 inserirArestas()

```
int inserirArestas (  
    Grafo * grafo)
```

Função para criar uma aresta de 2 antenas com a mesma função.

Parâmetros

<i>grafo</i>	grafo com antenas inseridas
--------------	-----------------------------

Retorna

1 = sucesso

4.4.2.24 lerFicheiroEscreverOutput()

```
int lerFicheiroEscreverOutput (  
    const char * nome_ficheiro)
```

Função para ler e escrever na consola o conteúdo do ficheiro binário.

Parâmetros

<i>nome_ficheiro</i>	nome do ficheiro binário
----------------------	--------------------------

Retorna

1 = sucesso

4.4.2.25 procurarAntenaId()

```
Antena * procurarAntenaId (  
    Grafo * grafo,  
    int id)
```

Função para procurar uma antena no grafo pelo id.

Parâmetros

<i>grafo</i>	grafo com as antenas
<i>id</i>	id da antena

Retorna

antena pedida

4.4.2.26 **removerDoCaminho()**

```
int removerDoCaminho (  
    Caminho * c)
```

Função para remover uma antena (id) de um caminho.

Parâmetros

<i>c</i>	caminho
----------	---------

Retorna

1 = sucesso

4.4.2.27 **segmentosSeCruzam()**

```
int segmentosSeCruzam (  
    int x1,  
    int y1,  
    int x2,  
    int y2,  
    int x3,  
    int y3,  
    int x4,  
    int y4)
```

Função para verificar se 2 segmentos se cruzam.

Parâmetros

<i>x1</i>	coordenada x do primeiro ponto
<i>y1</i>	coordenada y do primeiro ponto
<i>x2</i>	coordenada x do segundo ponto
<i>y2</i>	coordenada y do segundo ponto
<i>x3</i>	coordenada x do terceiro ponto
<i>y3</i>	coordenada y do terceiro ponto
<i>x4</i>	coordenada x do quarto ponto
<i>y4</i>	coordenada y do quarto ponto

Retorna

1 se os segmentos (x1,y1)-(x2,y2) e (x3,y3)-(x4,y4) se cruzam, 0 caso contrário

Índice

adicionarAoCamino

func.c, [25](#)

func.h, [13](#)

Antena, [5](#)

Aresta, [5](#)

bfslista, [6](#)

Caminho, [6](#)

carregarAntenas

func.c, [25](#)

func.h, [14](#)

criarAresta

func.c, [25](#)

func.h, [14](#)

criarGrafo

func.c, [25](#)

func.h, [14](#)

criarVertice

func.c, [26](#)

func.h, [14](#)

desenfileirar

func.c, [26](#)

func.h, [15](#)

detetarIntersecoes

func.c, [26](#)

func.h, [15](#)

dfs

func.c, [27](#)

func.h, [15](#)

dfsBacktracking

func.c, [27](#)

func.h, [16](#)

dfslista, [7](#)

encontrarCaminhoDFS

func.c, [27](#)

func.h, [16](#)

enfileirar

func.c, [28](#)

func.h, [16](#)

escreverAntenas

func.c, [28](#)

func.h, [17](#)

escreverBFS

func.c, [28](#)

func.h, [17](#)

escreverBloco

func.c, [29](#)

func.h, [17](#)

escreverCaminho

func.c, [29](#)

func.h, [18](#)

escreverDFS

func.c, [29](#)

func.h, [18](#)

escreverIntersecoes

func.c, [30](#)

func.h, [18](#)

execBfs

func.c, [30](#)

func.h, [19](#)

execDfs

func.c, [30](#)

func.h, [19](#)

Fila, [7](#)

filaVazia

func.c, [31](#)

func.h, [19](#)

freeMem

func.c, [31](#)

func.h, [20](#)

func.c

adicionarAoCamino, [25](#)

carregarAntenas, [25](#)

criarAresta, [25](#)

criarGrafo, [25](#)

criarVertice, [26](#)

desenfileirar, [26](#)

detetarIntersecoes, [26](#)

dfs, [27](#)

dfsBacktracking, [27](#)

encontrarCaminhoDFS, [27](#)

enfileirar, [28](#)

escreverAntenas, [28](#)

escreverBFS, [28](#)

escreverBloco, [29](#)

escreverCaminho, [29](#)

escreverDFS, [29](#)

escreverIntersecoes, [30](#)

execBfs, [30](#)

execDfs, [30](#)

filaVazia, [31](#)

freeMem, [31](#)

inicializarFila, [31](#)

inserirArestas, [32](#)

lerFicheiroEscreverOutput, [32](#)

procurarAntenald, [32](#)
removerDoCaminho, [33](#)
segmentosSeCruzam, [33](#)

func.h

- adicionarAoCamino, [13](#)
- carregarAntenas, [14](#)
- criarAresta, [14](#)
- criarGrafo, [14](#)
- criarVertice, [14](#)
- desenfileirar, [15](#)
- detetarIntersecoes, [15](#)
- dfs, [15](#)
- dfsBacktracking, [16](#)
- encontrarCaminhoDFS, [16](#)
- enfileirar, [16](#)
- escreverAntenas, [17](#)
- escreverBFS, [17](#)
- escreverBloco, [17](#)
- escreverCaminho, [18](#)
- escreverDFS, [18](#)
- escreverIntersecoes, [18](#)
- execBfs, [19](#)
- execDfs, [19](#)
- filaVazia, [19](#)
- freeMem, [20](#)
- inicializarFila, [20](#)
- inserirArestas, [20](#)
- lerFicheiroEscreverOutput, [21](#)
- procurarAntenald, [21](#)
- removerDoCaminho, [21](#)
- segmentosSeCruzam, [22](#)

func.h, [22](#)
src/func.c, [23](#)

Grafo, [8](#)

include/estruturas.h, [11](#)
include/func.h, [12](#), [22](#)

inicializarFila

- func.c, [31](#)
- func.h, [20](#)

inserirArestas

- func.c, [32](#)
- func.h, [20](#)

interlista, [8](#)

lerFicheiroEscreverOutput

- func.c, [32](#)
- func.h, [21](#)

NoCaminho, [9](#)

procurarAntenald

- func.c, [32](#)
- func.h, [21](#)

removerDoCaminho

- func.c, [33](#)
- func.h, [21](#)

segmentosSeCruzam

- func.c, [33](#)