

# Controle de fila com pré-cadastro

## V1.1

Generated by Doxygen 1.13.2



<b>1 README</b>	<b>1</b>
<b>2 Data Structure Index</b>	<b>3</b>
2.1 Data Structures . . . . .	3
<b>3 File Index</b>	<b>5</b>
3.1 File List . . . . .	5
<b>4 Data Structure Documentation</b>	<b>7</b>
4.1 FICHA Struct Reference . . . . .	7
4.1.1 Detailed Description . . . . .	7
4.1.2 Field Documentation . . . . .	7
4.1.2.1 id . . . . .	7
4.1.2.2 nome . . . . .	8
4.2 FILA Struct Reference . . . . .	8
4.2.1 Detailed Description . . . . .	8
4.2.2 Field Documentation . . . . .	8
4.2.2.1 fichas . . . . .	8
4.2.2.2 max . . . . .	8
4.2.2.3 primeiro . . . . .	9
4.2.2.4 tamanho . . . . .	9
4.2.2.5 ultimo . . . . .	9
4.3 ITERADOR Struct Reference . . . . .	9
4.3.1 Detailed Description . . . . .	9
4.3.2 Field Documentation . . . . .	9
4.3.2.1 estrutura . . . . .	9
4.3.2.2 posicao . . . . .	10
4.4 LISTA Struct Reference . . . . .	10
4.4.1 Detailed Description . . . . .	10
4.4.2 Field Documentation . . . . .	10
4.4.2.1 sentinela . . . . .	10
4.4.2.2 tamanho . . . . .	10
4.5 node Struct Reference . . . . .	11
4.5.1 Detailed Description . . . . .	11
4.5.2 Field Documentation . . . . .	11
4.5.2.1 anterior . . . . .	11
4.5.2.2 dado . . . . .	11
4.5.2.3 proximo . . . . .	11
<b>5 File Documentation</b>	<b>13</b>
5.1 Ficha.h File Reference . . . . .	13
5.1.1 Detailed Description . . . . .	13
5.2 Ficha.h . . . . .	13
5.3 Fila.c File Reference . . . . .	13

5.3.1 Detailed Description . . . . .	14
5.3.2 Function Documentation . . . . .	14
5.3.2.1 aumenta_fila() . . . . .	14
5.3.2.2 destruir_fila() . . . . .	14
5.3.2.3 fila_cheia() . . . . .	15
5.3.2.4 fila_vazia() . . . . .	15
5.3.2.5 inicia_fila() . . . . .	15
5.3.2.6 insere_fila() . . . . .	15
5.3.2.7 remover_ini() . . . . .	16
5.4 Fila.h File Reference . . . . .	16
5.4.1 Detailed Description . . . . .	17
5.4.2 Function Documentation . . . . .	17
5.4.2.1 aumenta_fila() . . . . .	17
5.4.2.2 destruir_fila() . . . . .	17
5.4.2.3 fila_cheia() . . . . .	17
5.4.2.4 fila_vazia() . . . . .	18
5.4.2.5 inicia_fila() . . . . .	18
5.4.2.6 insere_fila() . . . . .	18
5.4.2.7 remover_ini() . . . . .	19
5.5 Fila.h . . . . .	19
5.6 Lista.c File Reference . . . . .	19
5.6.1 Detailed Description . . . . .	20
5.6.2 Function Documentation . . . . .	20
5.6.2.1 adiciona_id() . . . . .	20
5.6.2.2 consulta() . . . . .	21
5.6.2.3 destruir_lista() . . . . .	21
5.6.2.4 ini_iterador() . . . . .	21
5.6.2.5 inicia_lista() . . . . .	21
5.6.2.6 lista_vazia() . . . . .	22
5.6.2.7 remove_ini_lista() . . . . .	22
5.6.2.8 tamanho() . . . . .	22
5.7 Lista.h File Reference . . . . .	23
5.7.1 Detailed Description . . . . .	24
5.7.2 Typedef Documentation . . . . .	24
5.7.2.1 ITEM . . . . .	24
5.7.2.2 PONTEIRO . . . . .	24
5.7.3 Function Documentation . . . . .	24
5.7.3.1 adiciona_id() . . . . .	24
5.7.3.2 consulta() . . . . .	25
5.7.3.3 destruir_lista() . . . . .	25
5.7.3.4 ini_iterador() . . . . .	26
5.7.3.5 inicia_lista() . . . . .	26

---

5.7.3.6 lista_vazia()	27
5.7.3.7 remove_ini_lista()	27
5.7.3.8 tamanho()	27
5.8 Lista.h	28
5.9 main.c File Reference	28
5.9.1 Detailed Description	29
5.9.2 Function Documentation	29
5.9.2.1 main()	29
5.10 README.md File Reference	29



# Chapter 1

## README

O projeto se baseia em uma lista de chamada baseada no id da pessoa, conseguido com o cadastro antecipado no site ou no local. Nele, você pode inserir a ficha de uma pessoa (nome e id). As já cadastradas entram em uma fila separada. O programa tem a função de organizar uma fila matendo uma proporção 2 para 1 em uma fila final de pessoas com e sem cadastro. Esse sistema pode ser usado em qualquer tipo de guichê que tenha cadastro antecipado como opção.





## Chapter 2

# Data Structure Index

### 2.1 Data Structures

Here are the data structures with brief descriptions:

<b>FICHA</b>	Estrutura que representa uma ficha com nome e ID . . . . .	7
<b>FILA</b>	Estrutura que representa uma fila de fichas . . . . .	8
<b>ITERADOR</b>	Estrutura do iterador para percorrer a lista . . . . .	9
<b>LISTA</b>	Estrutura que representa a lista duplamente encadeada . . . . .	10
<b>node</b>	Estrutura que representa um nó da lista . . . . .	11



## Chapter 3

# File Index

### 3.1 File List

Here is a list of all files with brief descriptions:

<b>Ficha.h</b>	Definição da estrutura <b>FICHA</b> (p. 7) para armazenar nome e ID de uma pessoa . . . . .	13
<b>Fila.c</b>	Implementação das operações para a fila de fichas . . . . .	13
<b>Fila.h</b>	Definição da estrutura e operações para a fila de fichas . . . . .	16
<b>Lista.c</b>	Implementação de uma lista duplamente encadeada circular com sentinela . . . . .	19
<b>Lista.h</b>	Definição da estrutura de uma lista duplamente encadeada circular com sentinela . . . . .	23
<b>main.c</b>	Programa principal para gerenciamento de filas e listas de cadastro . . . . .	28



## Chapter 4

# Data Structure Documentation

### 4.1 FICHA Struct Reference

Estrutura que representa uma ficha com nome e ID.

```
#include <Ficha.h>
```

#### Data Fields

- char **nome** [16]  
*Nome da pessoa.*
- unsigned **id**  
*Identificador único da pessoa.*

#### 4.1.1 Detailed Description

Estrutura que representa uma ficha com nome e ID.

Esta estrutura armazena o nome e o identificador único de uma pessoa.

#### 4.1.2 Field Documentation

##### 4.1.2.1 id

```
unsigned id
```

Identificador único da pessoa.

Este campo armazena o identificador único da pessoa, representado como um número inteiro não negativo.

#### 4.1.2.2 nome

```
char nome[16]
```

Nome da pessoa.

Este campo armazena o nome da pessoa com um tamanho máximo de 15 caracteres, já que o último caractere é reservado para o terminador nulo.

The documentation for this struct was generated from the following file:

- **Ficha.h**

## 4.2 FILA Struct Reference

Estrutura que representa uma fila de fichas.

```
#include <Fila.h>
```

### Data Fields

- **FICHA \* fichas**
- int **primeiro**
- int **ultimo**
- int **tamanho**
- int **max**

### 4.2.1 Detailed Description

Estrutura que representa uma fila de fichas.

### 4.2.2 Field Documentation

#### 4.2.2.1 fichas

```
FICHA* fichas
```

Vetor dinâmico para armazenar as fichas.

#### 4.2.2.2 max

```
int max
```

Capacidade máxima da fila.

#### 4.2.2.3 primeiro

```
int primeiro
```

Índice do primeiro elemento da fila.

#### 4.2.2.4 tamanho

```
int tamanho
```

Quantidade atual de elementos na fila.

#### 4.2.2.5 ultimo

```
int ultimo
```

Índice do último elemento da fila.

The documentation for this struct was generated from the following file:

- **Fila.h**

## 4.3 ITERADOR Struct Reference

Estrutura do iterador para percorrer a lista.

```
#include <Lista.h>
```

### Data Fields

- **PONTEIRO** posicao
- **LISTA \*** estrutura

### 4.3.1 Detailed Description

Estrutura do iterador para percorrer a lista.

### 4.3.2 Field Documentation

#### 4.3.2.1 estrutura

```
LISTA* estrutura
```

Ponteiro para a lista associada.

#### 4.3.2.2 posicao

**PONTEIRO** posicao

Posição atual do iterador.

The documentation for this struct was generated from the following file:

- **Lista.h**

## 4.4 LISTA Struct Reference

Estrutura que representa a lista duplamente encadeada.

```
#include <Lista.h>
```

### Data Fields

- **PONTEIRO** sentinela
- int tamanho

#### 4.4.1 Detailed Description

Estrutura que representa a lista duplamente encadeada.

#### 4.4.2 Field Documentation

##### 4.4.2.1 sentinela

**PONTEIRO** sentinela

Ponteiro para o nó sentinela.

##### 4.4.2.2 tamanho

int tamanho

Número de elementos na lista.

The documentation for this struct was generated from the following file:

- **Lista.h**



## 4.5 node Struct Reference

Estrutura que representa um nó da lista.

```
#include <Lista.h>
```

### Data Fields

- **FICHA** **dado**
- struct **node** \* **proximo**
- struct **node** \* **anterior**

### 4.5.1 Detailed Description

Estrutura que representa um nó da lista.

### 4.5.2 Field Documentation

#### 4.5.2.1 anterior

```
struct node * anterior
```

Ponteiros para o próximo e anterior nós.

#### 4.5.2.2 dado

**FICHA** **dado**

Dado armazenado no nó.

#### 4.5.2.3 proximo

```
struct node* proximo
```

The documentation for this struct was generated from the following file:

- **Lista.h**



## Chapter 5

# File Documentation

### 5.1 Ficha.h File Reference

Definição da estrutura **FICHA** (p. 7) para armazenar nome e ID de uma pessoa.

#### Data Structures

- struct **FICHA**

*Estrutura que representa uma ficha com nome e ID.*

#### 5.1.1 Detailed Description

Definição da estrutura **FICHA** (p. 7) para armazenar nome e ID de uma pessoa.

Este arquivo contém a definição da estrutura **FICHA** (p. 7), usada para armazenar informações sobre uma pessoa, incluindo o nome e o ID único.

### 5.2 Ficha.h

**Go to the documentation of this file.**

```
00001
00007
00008 #ifndef FICHA_H
00009 #define FICHA_H
00010
00017 typedef struct{
00024     char nome[16];
00025
00032     unsigned id;
00033 } FICHA;
00034
00035 #endif // FICHA_H
```

### 5.3 Fila.c File Reference

Implementação das operações para a fila de fichas.

```
#include "Fila.h"
```

## Functions

- **FILA \* inicia\_fila ( FILA \*p)**  
*Inicializa uma fila vazia.*
- **int aumenta\_fila ( FILA \*p)**  
*Aumenta a capacidade da fila.*
- **FICHA remover\_ini ( FILA \*p)**  
*Remove e retorna o primeiro elemento da fila.*
- **void destruir\_fila ( FILA \*p)**  
*Libera a memória ocupada pela fila.*
- **void insere\_fila ( FILA \*p, FICHA f)**  
*Insere uma ficha na fila.*
- **int fila\_vazia ( FILA \*p)**  
*Verifica se a fila está vazia.*
- **int fila\_cheia ( FILA \*p)**  
*Verifica se a fila está cheia.*

### 5.3.1 Detailed Description

Implementação das operações para a fila de fichas.

#### Author

pagfr

#### Date

25/02/2025

### 5.3.2 Function Documentation

#### 5.3.2.1 aumenta\_fila()

```
int aumenta_fila (  
    FILA * p)
```

Aumenta a capacidade da fila.

#### Parameters

<i>p</i>	Ponteiro para a fila.
----------	-----------------------

#### Returns

int Retorna 1 se bem-sucedido, 0 em caso de erro.

#### 5.3.2.2 destruir\_fila()

```
void destruir_fila (  
    FILA * p)
```

Libera a memória ocupada pela fila.

## Parameters

$p$	Ponteiro para a fila.
-----	-----------------------

**5.3.2.3 fila\_cheia()**

```
int fila_cheia (  
    FILA *  $p$ )
```

Verifica se a fila está cheia.

## Parameters

$p$	Ponteiro para a fila.
-----	-----------------------

## Returns

int Retorna 1 se estiver cheia, 0 caso contrário.

**5.3.2.4 fila\_vazia()**

```
int fila_vazia (  
    FILA *  $p$ )
```

Verifica se a fila está vazia.

## Parameters

$p$	Ponteiro para a fila.
-----	-----------------------

## Returns

int Retorna 1 se estiver vazia, 0 caso contrário.

**5.3.2.5 inicia\_fila()**

```
FILA * inicia_fila (  
    FILA *  $p$ )
```

Inicializa uma fila vazia.

Inicializa uma nova fila.

## Parameters

$p$	Ponteiro para a fila.
-----	-----------------------

## Returns

FILA\* Ponteiro para a fila inicializada.

**5.3.2.6 insere\_fila()**

```
void insere_fila (  
    FILA *  $p$ ,  
    FICHA  $f$ )
```

Insere uma ficha na fila.

## Parameters

<i>p</i>	Ponteiro para a fila.
<i>f</i>	Estrutura <b>FICHA</b> (p. 7) a ser inserida.

### 5.3.2.7 remover\_ini()

```
FICHA remover_ini (
    FILA * p)
```

Remove e retorna o primeiro elemento da fila.

## Parameters

<i>p</i>	Ponteiro para a fila.
----------	-----------------------

## Returns

**FICHA** (p. 7) A ficha removida.

## 5.4 Fila.h File Reference

Definição da estrutura e operações para a fila de fichas.

```
#include "Ficha.h"
#include <malloc.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

### Data Structures

- struct **FILA**  
*Estrutura que representa uma fila de fichas.*

### Functions

- **FILA** \* **inicia\_fila** ( **FILA** \*p)  
*Inicializa uma nova fila.*
- int **aumenta\_fila** ( **FILA** \*p)  
*Aumenta a capacidade da fila.*
- **FICHA** **remover\_ini** ( **FILA** \*p)  
*Remove e retorna o primeiro elemento da fila.*
- void **destruir\_fila** ( **FILA** \*p)  
*Libera a memória ocupada pela fila.*
- void **insere\_fila** ( **FILA** \*p, **FICHA** f)  
*Insere uma ficha na fila.*
- int **fila\_vazia** ( **FILA** \*p)  
*Verifica se a fila está vazia.*
- int **fila\_cheia** ( **FILA** \*p)  
*Verifica se a fila está cheia.*

### 5.4.1 Detailed Description

Definição da estrutura e operações para a fila de fichas.

#### Author

pagfr

#### Date

25/02/2025

### 5.4.2 Function Documentation

#### 5.4.2.1 aumenta\_fila()

```
int aumenta_fila (  
    FILA * p)
```

Aumenta a capacidade da fila.

#### Parameters

<i>p</i>	Ponteiro para a fila.
----------	-----------------------

#### Returns

int Retorna 1 se bem-sucedido, 0 em caso de erro.

#### 5.4.2.2 destruir\_fila()

```
void destruir_fila (  
    FILA * p)
```

Libera a memória ocupada pela fila.

#### Parameters

<i>p</i>	Ponteiro para a fila.
----------	-----------------------

#### 5.4.2.3 fila\_cheia()

```
int fila_cheia (  
    FILA * p)
```

Verifica se a fila está cheia.

**Parameters**

$p$	Ponteiro para a fila.
-----	-----------------------

**Returns**

int Retorna 1 se estiver cheia, 0 caso contrário.

**5.4.2.4 fila\_vazia()**

```
int fila_vazia (  
    FILA *  $p$ )
```

Verifica se a fila está vazia.

**Parameters**

$p$	Ponteiro para a fila.
-----	-----------------------

**Returns**

int Retorna 1 se estiver vazia, 0 caso contrário.

**5.4.2.5 inicia\_fila()**

```
FILA * inicia_fila (  
    FILA *  $p$ )
```

Inicializa uma nova fila.

**Parameters**

$p$	Ponteiro para a fila.
-----	-----------------------

**Returns**

**FILA**\* Ponteiro para a fila inicializada.

Inicializa uma nova fila.

**Parameters**

$p$	Ponteiro para a fila.
-----	-----------------------

**Returns**

**FILA**\* Ponteiro para a fila inicializada.

**5.4.2.6 insere\_fila()**

```
void insere_fila (  
    FILA *  $p$ ,  
    FICHA  $f$ )
```

Insere uma ficha na fila.



## Parameters

<i>p</i>	Ponteiro para a fila.
<i>f</i>	Estrutura <b>FICHA</b> (p. 7) a ser inserida.

## 5.4.2.7 remover\_ini()

```
FICHA remover_ini (
    FILA * p)
```

Remove e retorna o primeiro elemento da fila.

## Parameters

<i>p</i>	Ponteiro para a fila.
----------	-----------------------

## Returns

**FICHA** (p. 7) A ficha removida.

## 5.5 Fila.h

## Go to the documentation of this file.

```
00001
00002
00003 #ifndef FILA_H
00004 #define FILA_H
00005
00006 #include "Ficha.h"
00007 #include <malloc.h>
00008 #include <stdio.h>
00009 #include <stdlib.h>
00010 #include <string.h>
00011
00012 typedef struct {
00013     FICHA *fichas;
00014     int primeiro;
00015     int ultimo;
00016     int tamanho;
00017     int max;
00018 } FILA;
00019
00020 FILA *inicia_fila(FILA *p);
00021
00022 int aumenta_fila(FILA *p);
00023
00024 FICHA remover_ini(FILA *p);
00025
00026 void destruir_fila(FILA *p);
00027
00028 void insere_fila(FILA *p, FICHA f);
00029
00030 int fila_vazia(FILA *p);
00031
00032 int fila_cheia(FILA *p);
00033
00034 #endif // FILA_H
```

## 5.6 Lista.c File Reference

Implementação de uma lista duplamente encadeada circular com sentinela.

```
#include "Lista.h"
```

## Functions

- **LISTA \* inicia\_lista ( LISTA \*p)**  
*Inicializa a lista duplamente encadeada com sentinela.*
- **int adiciona\_id ( LISTA \*p, FICHA f)**  
*Adiciona um novo ID na lista.*
- **int consulta ( LISTA \*p, FICHA f)**  
*Consulta e remove um elemento da lista pelo ID.*
- **ITERADOR ini\_iterador ( LISTA \*p)**  
*Inicializa um iterador para a lista.*
- **int tamanho ( LISTA \*p)**  
*Retorna o tamanho da lista.*
- **int lista\_vazia ( LISTA \*p)**  
*Verifica se a lista está vazia.*
- **void destruir\_lista ( LISTA \*p)**  
*Libera a memória ocupada pela lista.*
- **void remove\_ini\_lista ( LISTA \*p)**  
*Remove o primeiro elemento da lista (usado apenas na destruição).*

### 5.6.1 Detailed Description

Implementação de uma lista duplamente encadeada circular com sentinela.

#### Author

pagfr

#### Date

25/02/2025

### 5.6.2 Function Documentation

#### 5.6.2.1 adiciona\_id()

```
int adiciona_id (
    LISTA * p,
    FICHA f)
```

Adiciona um novo ID na lista.

Adiciona um ID à lista.

#### Parameters

<i>p</i>	Ponteiro para a lista.
<i>f</i>	Estrutura do tipo <b>FICHA</b> (p. 7) contendo os dados.

#### Returns

int Retorna 1 se bem-sucedido, 0 em caso de erro.

### 5.6.2.2 consulta()

```
int consulta (  
    LISTA * p,  
    FICHA f)
```

Consulta e remove um elemento da lista pelo ID.

Consulta um ID na lista e remove se encontrado.

#### Parameters

<i>p</i>	Ponteiro para a lista.
<i>f</i>	Estrutura <b>FICHA</b> (p. 7) com o ID a ser removido.

#### Returns

int Retorna 1 se encontrou e removeu, 0 caso contrário.

### 5.6.2.3 destruir\_lista()

```
void destruir_lista (  
    LISTA * p)
```

Libera a memória ocupada pela lista.

Libera toda a memória ocupada pela lista.

#### Parameters

<i>p</i>	Ponteiro para a lista.
----------	------------------------

### 5.6.2.4 ini\_iterador()

```
ITERADOR ini_iterador (  
    LISTA * p)
```

Inicializa um iterador para a lista.

Inicializa um iterador para percorrer a lista.

#### Parameters

<i>p</i>	Ponteiro para a lista.
----------	------------------------

#### Returns

**ITERADOR** (p. 9) Retorna um iterador inicializado.

### 5.6.2.5 inicia\_lista()

```
LISTA * inicia_lista (  
    LISTA * p)
```

Inicializa a lista duplamente encadeada com sentinela.

Inicializa uma nova lista.

**Parameters**

<i>p</i>	Ponteiro para a lista (inicialmente nulo).
----------	--

**Returns**

LISTA\* Ponteiro para a lista inicializada.

**5.6.2.6 lista\_vazia()**

```
int lista_vazia (  
    LISTA * p)
```

Verifica se a lista está vazia.

**Parameters**

<i>p</i>	Ponteiro para a lista.
----------	------------------------

**Returns**

int Retorna 1 se estiver vazia, 0 caso contrário.

**5.6.2.7 remove\_ini\_lista()**

```
void remove_ini_lista (  
    LISTA * p)
```

Remove o primeiro elemento da lista (usado apenas na destruição).

Remove o primeiro elemento da lista (usado na destruição).

**Parameters**

<i>p</i>	Ponteiro para a lista.
----------	------------------------

**5.6.2.8 tamanho()**

```
int tamanho (  
    LISTA * p)
```

Retorna o tamanho da lista.

**Parameters**

<i>p</i>	Ponteiro para a lista.
----------	------------------------

**Returns**

int O número de elementos na lista.

## 5.7 Lista.h File Reference

Definição da estrutura de uma lista duplamente encadeada circular com sentinela.

```
#include "Ficha.h"
#include <malloc.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

### Data Structures

- struct **node**  
*Estrutura que representa um nó da lista.*
- struct **LISTA**  
*Estrutura que representa a lista duplamente encadeada.*
- struct **ITERADOR**  
*Estrutura do iterador para percorrer a lista.*

### Typedefs

- typedef struct **node** **ITEM**  
*Estrutura que representa um nó da lista.*
- typedef **ITEM** \* **PONTEIRO**

### Functions

- **LISTA** \* **inicia\_lista** ( **LISTA** \*p)  
*Inicializa uma nova lista.*
- int **adiciona\_id** ( **LISTA** \*p, **FICHA** c)  
*Adiciona um ID à lista.*
- int **consulta** ( **LISTA** \*p, **FICHA** f)  
*Consulta um ID na lista e remove se encontrado.*
- **ITERADOR** **ini\_iterador** ( **LISTA** \*p)  
*Inicializa um iterador para percorrer a lista.*
- int **tamanho** ( **LISTA** \*p)  
*Retorna o tamanho da lista.*
- int **lista\_vazia** ( **LISTA** \*p)  
*Verifica se a lista está vazia.*
- void **destruir\_lista** ( **LISTA** \*p)  
*Libera toda a memória ocupada pela lista.*
- void **remove\_ini\_lista** ( **LISTA** \*p)  
*Remove o primeiro elemento da lista (usado na destruição).*

### 5.7.1 Detailed Description

Definição da estrutura de uma lista duplamente encadeada circular com sentinela.

Author

pagfr

Date

25/02/2025

### 5.7.2 Typedef Documentation

#### 5.7.2.1 ITEM

```
typedef struct node ITEM
```

Estrutura que representa um nó da lista.

#### 5.7.2.2 PONTEIRO

```
typedef ITEM* PONTEIRO
```

### 5.7.3 Function Documentation

#### 5.7.3.1 adiciona\_id()

```
int adiciona_id (  
    LISTA * p,  
    FICHA f)
```

Adiciona um ID à lista.

Parameters

<i>p</i>	Ponteiro para a lista.
<i>c</i>	Estrutura do tipo <b>FICHA</b> (p. 7) contendo os dados.

Returns

int Retorna 1 se bem-sucedido, 0 em caso de erro.

Adiciona um ID à lista.

## Parameters

<i>p</i>	Ponteiro para a lista.
<i>f</i>	Estrutura do tipo <b>FICHA</b> (p. 7) contendo os dados.

## Returns

int Retorna 1 se bem-sucedido, 0 em caso de erro.

**5.7.3.2 consulta()**

```
int consulta (  
    LISTA * p,  
    FICHA f)
```

Consulta um ID na lista e remove se encontrado.

## Parameters

<i>p</i>	Ponteiro para a lista.
<i>f</i>	Estrutura <b>FICHA</b> (p. 7) com o ID a ser removido.

## Returns

int Retorna 1 se encontrou e removeu, 0 caso contrário.

Consulta um ID na lista e remove se encontrado.

## Parameters

<i>p</i>	Ponteiro para a lista.
<i>f</i>	Estrutura <b>FICHA</b> (p. 7) com o ID a ser removido.

## Returns

int Retorna 1 se encontrou e removeu, 0 caso contrário.

**5.7.3.3 destruir\_lista()**

```
void destruir_lista (  
    LISTA * p)
```

Libera toda a memória ocupada pela lista.

## Parameters

<i>p</i>	Ponteiro para a lista.
----------	------------------------

Libera toda a memória ocupada pela lista.

**Parameters**

$p$	Ponteiro para a lista.
-----	------------------------

**5.7.3.4 ini\_iterador()**

```
ITERADOR ini_iterador (  
    LISTA *  $p$ )
```

Inicializa um iterador para percorrer a lista.

**Parameters**

$p$	Ponteiro para a lista.
-----	------------------------

**Returns**

**ITERADOR** (p. 9) Iterador inicializado.

Inicializa um iterador para percorrer a lista.

**Parameters**

$p$	Ponteiro para a lista.
-----	------------------------

**Returns**

**ITERADOR** (p. 9) Retorna um iterador inicializado.

**5.7.3.5 inicia\_lista()**

```
LISTA * inicia_lista (  
    LISTA *  $p$ )
```

Inicializa uma nova lista.

**Parameters**

$p$	Ponteiro para a lista.
-----	------------------------

**Returns**

**LISTA\*** Ponteiro para a lista inicializada.

Inicializa uma nova lista.



**Parameters**

<i>p</i>	Ponteiro para a lista (inicialmente nulo).
----------	--

**Returns**

LISTA\* Ponteiro para a lista inicializada.

**5.7.3.6 lista\_vazia()**

```
int lista_vazia (  
                LISTA * p)
```

Verifica se a lista está vazia.

**Parameters**

<i>p</i>	Ponteiro para a lista.
----------	------------------------

**Returns**

int Retorna 1 se estiver vazia, 0 caso contrário.

**5.7.3.7 remove\_ini\_lista()**

```
void remove_ini_lista (  
                LISTA * p)
```

Remove o primeiro elemento da lista (usado na destruição).

**Parameters**

<i>p</i>	Ponteiro para a lista.
----------	------------------------

Remove o primeiro elemento da lista (usado na destruição).

**Parameters**

<i>p</i>	Ponteiro para a lista.
----------	------------------------

**5.7.3.8 tamanho()**

```
int tamanho (  
            LISTA * p)
```

Retorna o tamanho da lista.

**Parameters**

<i>p</i>	Ponteiro para a lista.
----------	------------------------

**Returns**

int O número de elementos na lista.

## 5.8 Lista.h

**Go to the documentation of this file.**

```

00001
00007
00008 #ifndef LISTA_H
00009 #define LISTA_H
00010
00011 #include "Ficha.h"
00012 #include <malloc.h>
00013 #include <stdio.h>
00014 #include <stdlib.h>
00015 #include <string.h>
00016
00020 typedef struct node {
00021     FICHA dado;
00022     struct node *proximo, *anterior;
00023 } ITEM;
00024
00025 typedef ITEM *PONTEIRO;
00026
00030 typedef struct {
00031     PONTEIRO sentinela;
00032     int tamanho;
00033 } LISTA;
00034
00038 typedef struct {
00039     PONTEIRO posicao;
00040     LISTA *estrutura;
00041 } ITERADOR;
00042
00048 LISTA *inicia_lista(LISTA *p);
00049
00056 int adiciona_id(LISTA *p, FICHA c);
00057
00064 int consulta(LISTA *p, FICHA f);
00065
00071 ITERADOR ini_iterador(LISTA *p);
00072
00078 int tamanho(LISTA *p);
00079
00085 int lista_vazia(LISTA *p);
00086
00091 void destruir_lista(LISTA *p);
00092
00097 void remove_ini_lista(LISTA *p);
00098
00099 #endif // LISTA_H

```

## 5.9 main.c File Reference

Programa principal para gerenciamento de filas e listas de cadastro.

```

#include "Ficha.h"
#include "Fila.h"
#include "Lista.h"
#include <ctype.h>
#include <stdio.h>
#include <stdlib.h>

```

## Functions

- `int main ()`  
*Função principal do programa.*

### 5.9.1 Detailed Description

Programa principal para gerenciamento de filas e listas de cadastro.

#### Author

pagfr

#### Date

26/02/2025

### 5.9.2 Function Documentation

#### 5.9.2.1 main()

```
int main ()
```

Função principal do programa.

Gerencia filas e listas de cadastro, permitindo adicionar IDs à fila, chamar pessoas da fila e cadastrar novos IDs na lista.

#### Returns

`int` Retorna 0 ao finalizar a execução.

Caso 1 - Adiciona uma pessoa à fila.

Caso 2 - Remove uma pessoa da fila e a exibe.

Caso 3 - Cadastra um novo ID na lista.

Caso -1 - Finaliza o programa e libera memória.

## 5.10 README.md File Reference

