

Programaç?o - EGI

Generated by Doxygen 1.9.3



<b>1 Bug List</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 Contacto Struct Reference . . . . .	7
4.1.1 Detailed Description . . . . .	7
4.1.2 Field Documentation . . . . .	7
4.1.2.1 desc . . . . .	7
4.1.2.2 valor . . . . .	8
4.2 ListaContactos Struct Reference . . . . .	8
4.2.1 Detailed Description . . . . .	8
4.2.2 Field Documentation . . . . .	8
4.2.2.1 contacto . . . . .	8
4.2.2.2 proxContacto . . . . .	8
4.3 ListaPessoa Struct Reference . . . . .	9
4.3.1 Detailed Description . . . . .	9
4.3.2 Field Documentation . . . . .	9
4.3.2.1 fichaPessoa . . . . .	9
4.3.2.2 listaContactos . . . . .	9
4.3.2.3 proxPessoa . . . . .	10
4.4 Pessoa Struct Reference . . . . .	10
4.4.1 Detailed Description . . . . .	10
4.4.2 Field Documentation . . . . .	10
4.4.2.1 nc . . . . .	10
4.4.2.2 nome . . . . .	11
4.5 TotalInformacaoPessoa Struct Reference . . . . .	11
4.5.1 Detailed Description . . . . .	11
4.5.2 Field Documentation . . . . .	11
4.5.2.1 contacto . . . . .	11
4.5.2.2 nc . . . . .	11
<b>5 File Documentation</b>	<b>13</b>
5.1 D:/Temp/PROG-EGI/Aulas/GereContactos/Contactos.c File Reference . . . . .	13
5.1.1 Detailed Description . . . . .	13
5.1.2 Function Documentation . . . . .	14
5.1.2.1 CriaContacto() . . . . .	14
5.1.2.2 CriaNodoListaContactos() . . . . .	14
5.1.2.3 InsereContactoListaContactos() . . . . .	14
5.1.2.4 MostraContactos() . . . . .	15

5.2 Contactos.c	15
5.3 D:/Temp/PROG-EGI/Aulas/GereContactos/Contatos.h File Reference	15
5.3.1 Macro Definition Documentation	16
5.3.1.1 M	16
5.3.1.2 N	16
5.3.2 Typedef Documentation	16
5.3.2.1 Contacto	16
5.3.2.2 ListaContactos	17
5.3.3 Function Documentation	17
5.3.3.1 CriaContacto()	17
5.3.3.2 CriaNodoListaContactos()	17
5.3.3.3 InsereContactoListaContactos()	17
5.3.3.4 MostraContactos()	18
5.4 Contatos.h	18
5.5 D:/Temp/PROG-EGI/Aulas/GereContactos/Main.c File Reference	18
5.5.1 Function Documentation	18
5.5.1.1 main()	19
5.6 Main.c	19
5.7 D:/Temp/PROG-EGI/Aulas/GereContactos/Pessoa.h File Reference	19
5.7.1 Detailed Description	21
5.7.2 Macro Definition Documentation	21
5.7.2.1 M	21
5.7.3 Typedef Documentation	21
5.7.3.1 CONTROLERROR	21
5.7.3.2 ListaPessoa	21
5.7.3.3 Pessoa	22
5.7.3.4 TodaInformacaoPessoa	22
5.7.4 Enumeration Type Documentation	22
5.7.4.1 CONTROLERROR	22
5.7.5 Function Documentation	22
5.7.5.1 CriaLista()	22
5.7.5.2 CriaNodoListaPessoas()	23
5.7.5.3 CriaPessoa()	23
5.7.5.4 GetAllContacts()	23
5.7.5.5 GetAllPessoas()	23
5.7.5.6 GetData()	24
5.7.5.7 InsereContactoPessoa()	24
5.7.5.8 InserePessoaListaPessoas()	24
5.7.5.9 MostraContactosPessoa()	25
5.7.5.10 MostraTodasPessoas()	25
5.7.5.11 ProcuraPessoa()	25
5.7.5.12 SaveAll()	25

---

5.7.5.13 SavePessoas()	26
5.8 Pessoa.h	26
5.9 D:/Temp/PROG-EGI/Aulas/GereContactos/Pessoas.c File Reference	27
5.9.1 Detailed Description	28
5.9.2 Macro Definition Documentation	28
5.9.2.1 MAX	28
5.9.3 Function Documentation	28
5.9.3.1 CriaLista()	29
5.9.3.2 CriaNodoListaPessoas()	29
5.9.3.3 CriaPessoa()	29
5.9.3.4 GetAllContacts()	30
5.9.3.5 GetAllPessoas()	30
5.9.3.6 GetData()	30
5.9.3.7 InsereContactoPessoa()	31
5.9.3.8 InserePessoaListaPessoas()	31
5.9.3.9 MostraContactosPessoa()	31
5.9.3.10 MostraTodasPessoas()	31
5.9.3.11 ProcuraPessoa()	32
5.9.3.12 SaveAll()	32
5.9.3.13 SavePessoas()	32
5.10 Pessoas.c	33
<b>Index</b>	<b>37</b>



# Chapter 1

## Bug List

File [Contactos.c](#)

No known bugs.

File [Pessoa.h](#)

No known bugs.

File [Pessoas.c](#)

No known bugs.





## Chapter 2

# Data Structure Index

### 2.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">Contacto</a>	7
<a href="#">ListaContactos</a>	8
<a href="#">ListaPessoa</a>	
Lista de Pessoas Informação sobre a <a href="#">Pessoa</a> , os seus contactos e apontador para outra <a href="#">Pessoa</a>	9
<a href="#">Pessoa</a>	
Informação de uma pessoa	10
<a href="#">TotalInformacaoPessoa</a>	11



## Chapter 3

# File Index

### 3.1 File List

Here is a list of all files with brief descriptions:

D:/Temp/PROG-EGI/Aulas/GereContactos/Contactos.c	
Lista Ligadas Simples (versão 1)	13
D:/Temp/PROG-EGI/Aulas/GereContactos/Contatos.h	15
D:/Temp/PROG-EGI/Aulas/GereContactos/Main.c	18
D:/Temp/PROG-EGI/Aulas/GereContactos/Pessoa.h	
Lista Ligadas Simples (versão 1)	19
D:/Temp/PROG-EGI/Aulas/GereContactos/Pessoas.c	
Lista Ligadas Simples (versão 1)	27



## Chapter 4

# Data Structure Documentation

### 4.1 Contacto Struct Reference

```
#include <Contatos.h>
```

#### Data Fields

- char [desc](#) [M]
- char [valor](#) [N]

#### 4.1.1 Detailed Description

Definition at line [21](#) of file [Contatos.h](#).

#### 4.1.2 Field Documentation

##### 4.1.2.1 desc

```
char desc[M]
```

designação do contacto. ex: Telefone

Definition at line [22](#) of file [Contatos.h](#).

#### 4.1.2.2 valor

```
char valor[N]
```

Valor do contacto. ex: 88997788

Definition at line 23 of file [Contatos.h](#).

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

- D:/Temp/PROG-EGI/Aulas/GereContactos/[Contatos.h](#)

## 4.2 ListaContactos Struct Reference

```
#include <Contatos.h>
```

### Data Fields

- struct [Contacto](#) contacto
- struct [ListaContactos](#) \* proxContacto

### 4.2.1 Detailed Description

Definition at line 26 of file [Contatos.h](#).

### 4.2.2 Field Documentation

#### 4.2.2.1 contacto

```
struct Contacto contacto
```

Definition at line 27 of file [Contatos.h](#).

#### 4.2.2.2 proxContacto

```
struct ListaContactos* proxContacto
```

Definition at line 28 of file [Contatos.h](#).

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

- D:/Temp/PROG-EGI/Aulas/GereContactos/[Contatos.h](#)

## 4.3 ListaPessoa Struct Reference

Lista de Pessoas Informação sobre a [Pessoa](#), os seus contactos e apontador para outra [Pessoa](#).

```
#include <Pessoa.h>
```

### Data Fields

- struct [Pessoa](#) fichaPessoa
- struct [ListaContactos](#) \* listaContactos
- struct [ListaPessoa](#) \* proxPessoa

#### 4.3.1 Detailed Description

Lista de Pessoas Informação sobre a [Pessoa](#), os seus contactos e apontador para outra [Pessoa](#).

Definition at line 31 of file [Pessoa.h](#).

#### 4.3.2 Field Documentation

##### 4.3.2.1 fichaPessoa

```
struct Pessoa fichaPessoa
```

toda a informação da [Pessoa](#)

Definition at line 32 of file [Pessoa.h](#).

##### 4.3.2.2 listaContactos

```
struct ListaContactos* listaContactos
```

Todos os contactos da [Pessoa](#)

Definition at line 33 of file [Pessoa.h](#).

#### 4.3.2.3 proxPessoa

```
struct ListaPessoa* proxPessoa
```

Ligação a outra pessoa

Definition at line 34 of file [Pessoa.h](#).

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

- D:/Temp/PROG-EGI/Aulas/GereContactos/[Pessoa.h](#)

## 4.4 Pessoa Struct Reference

Informação de uma pessoa.

```
#include <Pessoa.h>
```

### Data Fields

- int [nc](#)
- char [nome](#) [M]

#### 4.4.1 Detailed Description

Informação de uma pessoa.

Definition at line 22 of file [Pessoa.h](#).

#### 4.4.2 Field Documentation

##### 4.4.2.1 nc

```
int nc
```

Número de Contribuinte

Definition at line 23 of file [Pessoa.h](#).



#### 4.4.2.2 nome

```
char nome[M]
```

Nome da [Pessoa](#)

Definition at line 24 of file [Pessoa.h](#).

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

- D:/Temp/PROG-EGI/Aulas/GereContactos/[Pessoa.h](#)

## 4.5 TotalInformacaoPessoa Struct Reference

```
#include <Pessoa.h>
```

### Data Fields

- int [nc](#)
- [Contacto](#) [contacto](#)

### 4.5.1 Detailed Description

Preservar dados em ficheiro

Definition at line 73 of file [Pessoa.h](#).

### 4.5.2 Field Documentation

#### 4.5.2.1 contacto

[Contacto](#) [contacto](#)

[Contacto](#) da pessoa

Definition at line 75 of file [Pessoa.h](#).

#### 4.5.2.2 nc

```
int nc
```

Numero de Contribuinte da pessoa

Definition at line 74 of file [Pessoa.h](#).

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

- D:/Temp/PROG-EGI/Aulas/GereContactos/[Pessoa.h](#)



## Chapter 5

# File Documentation

### 5.1 D:/Temp/PROG-EGI/Aulas/GereContactos/Contactos.c File Reference

Lista Ligadas Simples (versão 1)

```
#include "Contatos.h"
#include "Pessoa.h"
#include <stdbool.h>
```

#### Functions

- [Contacto](#) \* [CriaContacto](#) (char \*desc, char \*valor)
- [ListaContactos](#) \* [CriaNodoListaContactos](#) ([Contacto](#) \*c)  
*Cria novo nodo para a Lista de Contactos Copia para o nodo da lista a informação de um contacto.*
- [ListaContactos](#) \* [InsereContactoListaContactos](#) ([ListaContactos](#) \*h, [Contacto](#) \*novoContacto)  
*Insere um novo contacto na lista.*
- void [MostraContactos](#) ([ListaContactos](#) \*h)

#### 5.1.1 Detailed Description

Lista Ligadas Simples (versão 1)

Author

lufer

Date

2022

Metodos para manipular uma Lista Ligada Simples de Contactos

**Bug** No known bugs.

Definition in file [Contactos.c](#).

## 5.1.2 Function Documentation

### 5.1.2.1 CriaContacto()

```
Contacto * CriaContacto (
    char * desc,
    char * valor )
```

#### Parameters

<i>desc</i>	
<i>valor</i>	

#### Returns

Definition at line 22 of file [Contactos.c](#).

### 5.1.2.2 CriaNodoListaContactos()

```
ListaContactos * CriaNodoListaContactos (
    Contacto * c )
```

Cria novo nodo para a Lista de Contactos Copia para o nodo da lista a informação de um contacto.

#### Parameters

in	<i>c</i>	Novo contacto
out	<i>Apontador</i>	para nodo criado

Definition at line 35 of file [Contactos.c](#).

### 5.1.2.3 InsereContactoListaContactos()

```
ListaContactos * InsereContactoListaContactos (
    ListaContactos * h,
    Contacto * novoContacto )
```

Insere um novo contacto na lista.

Definition at line 46 of file [Contactos.c](#).

### 5.1.2.4 MostraContactos()

```
void MostraContactos (
    ListaContactos * h )
```

Definition at line 61 of file [Contactos.c](#).

## 5.2 Contactos.c

[Go to the documentation of this file.](#)

```
00001
00011 #include "Contatos.h"
00012 #include "Pessoa.h"
00013 #include <stdbool.h>
00014
00022 Contacto* CriaContacto(char* desc, char* valor) {
00023     Contacto* aux = (Contacto*)calloc(1, sizeof(Contacto));
00024     strcpy(aux->desc, desc);
00025     strcpy(aux->valor, valor);
00026     return aux;
00027 }
00028
00035 ListaContactos* CriaNodoListaContactos(Contacto* c) {
00036     ListaContactos* novo = (ListaContactos*)calloc(1, sizeof(ListaContactos));
00037     strcpy(novo->contacto.desc, c->desc);
00038     strcpy(novo->contacto.valor, c->valor);
00039     novo->proxContacto = NULL;
00040     return novo;
00041 }
00042
00046 ListaContactos* InsereContactoListaContactos(ListaContactos* h, Contacto* novoContacto) {
00047     if (novoContacto == NULL) return h; //se novo está vazio
00048     //Cria novo nodo da lista de contactos
00049     ListaContactos* novo = CriaNodoListaContactos(novoContacto);
00050     if (h == NULL) h = novo; //se lista está vazia
00051     else {
00052         //Assumir que se insere sempre no inicio
00053         novo->proxContacto = h;
00054         h = novo;
00055     }
00056     return h;
00057 }
00058
00061 void MostraContactos(ListaContactos* h) {
00062     ListaContactos* aux = h;
00063     while (aux) {
00064         printf("Contacto: %s - Valor: %s\n", aux->contacto.desc, aux->contacto.valor);
00065         aux = aux->proxContacto;
00066     }
00067 }
00068 }
00069
```

## 5.3 D:/Temp/PROG-EGI/Aulas/GereContactos/Contatos.h File Reference

```
#include <stdio.h>
#include <stdbool.h>
```

### Data Structures

- struct [Contacto](#)
- struct [ListaContactos](#)

## Macros

- `#define M 40`
- `#define N 40`

## Typedefs

- `typedef struct Contacto Contacto`
- `typedef struct ListaContactos ListaContactos`

## Functions

- `Contacto * CriaContacto (char *desc, char *valor)`
- `void MostraContactos (ListaContactos *h)`
- `ListaContactos * InsereContactoListaContactos (ListaContactos *h, Contacto *novo)`  
*Insere um novo contacto na lista.*
- `ListaContactos * CriaNodoListaContactos (Contacto *c)`  
*Cria novo nodo para a Lista de Contactos Copia para o nodo da lista a informação de um contacto.*

### 5.3.1 Macro Definition Documentation

#### 5.3.1.1 M

```
#define M 40
```

Definition at line 16 of file [Contatos.h](#).

#### 5.3.1.2 N

```
#define N 40
```

Definition at line 17 of file [Contatos.h](#).

### 5.3.2 Typedef Documentation

#### 5.3.2.1 Contacto

```
typedef struct Contacto Contacto
```

### 5.3.2.2 ListaContactos

```
typedef struct ListaContactos ListaContactos
```

## 5.3.3 Function Documentation

### 5.3.3.1 CriaContacto()

```
Contacto * CriaContacto (
    char * desc,
    char * valor )
```

#### Parameters

<i>desc</i>	
<i>valor</i>	

#### Returns

Definition at line 22 of file [Contactos.c](#).

### 5.3.3.2 CriaNodoListaContactos()

```
ListaContactos * CriaNodoListaContactos (
    Contacto * c )
```

Cria novo nodo para a Lista de Contactos Copia para o nodo da lista a informação de um contacto.

#### Parameters

in	<i>c</i>	Novo contacto
out	<i>Apontador</i>	para nodo criado

Definition at line 35 of file [Contactos.c](#).

### 5.3.3.3 InsereContactoListaContactos()

```
ListaContactos * InsereContactoListaContactos (
    ListaContactos * h,
    Contacto * novo )
```

Insere um novo contacto na lista.

Definition at line 46 of file [Contactos.c](#).

#### 5.3.3.4 MostraContactos()

```
void MostraContactos (
    ListaContactos * h )
```

Definition at line 61 of file [Contactos.c](#).

## 5.4 Contatos.h

[Go to the documentation of this file.](#)

```
00001
00010 #pragma once
00011 #pragma warning( disable : 4996 ) //evita MSG ERROS: _CRT_SECURE_NO_WARNINGS
00012
00013 #include <stdio.h>
00014 #include <stdbool.h>
00015
00016 #define M 40
00017 #define N 40
00018
00019 #pragma region GereContactos
00020
00021 typedef struct Contacto {
00022     char desc[M];
00023     char valor[N];
00024 }Contacto;
00025
00026 typedef struct ListaContactos {
00027     struct Contacto contacto;
00028     struct ListaContactos* proxContacto;
00029 }ListaContactos;
00030
00031
00032 Contacto* CriaContacto(char* desc, char* valor);
00033 void MostraContactos(ListaContactos* h);
00034 ListaContactos* InsereContactoListaContactos(ListaContactos* h, Contacto* novo);
00035 ListaContactos* CriaNodeListaContactos(Contacto* c);
00036
00037 #pragma endregion
```

## 5.5 D:/Temp/PROG-EGI/Aulas/GereContactos/Main.c File Reference

```
#include "Contatos.h"
#include "Pessoa.h"
```

### Functions

- int [main](#) ()

#### 5.5.1 Function Documentation



### 5.5.1.1 main()

```
int main ( )
```

Definition at line 13 of file [Main.c](#).

## 5.6 Main.c

[Go to the documentation of this file.](#)

```
00001
00010 #include "Contatos.h"
00011 #include "Pessoa.h"
00012
00013 int main() {
00014
00015     ListaPessoa* inicio = NULL;
00016     inicio= GetData("Dados.txt");
00017
00018     if (inicio == NULL) { //Se não existem dados de entrada, insere manualmente
00019         Contacto* c1 = CriaContacto("Telef", "253123321");
00020         ListaContactos* contactos = NULL;
00021         contactos = InsereContactoListaContactos(contactos, c1);
00022         //MostraContactos(contactos);
00023
00024         Pessoa* p = CriaPessoa("PP", 12345);
00025         inicio = InserePessoaListaPessoas(inicio, p);
00026
00027         Pessoa* p2 = CriaPessoa("JJ", 4321);
00028         inicio = InserePessoaListaPessoas(inicio, p2);
00029         inicio = InsereContactoPessoa(inicio, c1, 12345);
00030
00031         Contacto* c2 = CriaContacto("Email", "aaa@bbb.pt");
00032         inicio = InsereContactoPessoa(inicio, c2, 4321);
00033     }
00034
00035     MostraTodasPessoas(inicio);
00036     MostraContactosPessoa(inicio, 1);
00037     MostraContactosPessoa(inicio, 123);
00038
00039     CONTROLEERROR aux = SavePessoas(inicio, "Pessoas.dat");
00040     if (aux == NO_FILE) {
00041         puts("Não foi possível gravar...");
00042     }
00043
00044     SaveAll(inicio, "All.dat");
00045
00046     MostraTodasPessoas(inicio);
00047
00048     inicio = NULL; //ATENÇÃO: não é suficiente NULL...deve destruir-se a lista
00049     inicio = GetAllPessoas("Pessoas.dat");
00050     inicio = GetAllContacts("All.dat", inicio);
00051
00052
00053
00054
00055     MostraTodasPessoas(inicio);
00056     MostraContactosPessoa(inicio, 123);
00057 }
```

## 5.7 D:/Temp/PROG-EGI/Aulas/GereContactos/Pessoa.h File Reference

Lista Ligadas Simples (versão 1)

```
#include <stdio.h>
#include "Contatos.h"
#include <stdbool.h>
```

## Data Structures

- struct [Pessoa](#)  
*Informação de uma pessoa.*
- struct [ListaPessoa](#)  
*Lista de Pessoas Informação sobre a [Pessoa](#), os seus contactos e apontador para outra [Pessoa](#).*
- struct [TotalInformacaoPessoa](#)

## Macros

- `#define M 20`

## Typedefs

- typedef struct [Pessoa](#) [Pessoa](#)  
*Informação de uma pessoa.*
- typedef struct [ListaPessoa](#) [ListaPessoa](#)  
*Lista de Pessoas Informação sobre a [Pessoa](#), os seus contactos e apontador para outra [Pessoa](#).*
- typedef enum [CONTROLERROR](#) [CONTROLERROR](#)
- typedef struct [TotalInformacaoPessoa](#) [TotalInformacaoPessoa](#)

## Enumerations

- enum [CONTROLERROR](#) { [NO\\_ERROR](#) = 1 , [NO\\_FILE](#) =45 , [PRESIDENT](#) =100 , [NO\\_DATA](#) =2 }

## Functions

- [Pessoa \\*](#) [CriaLista](#) ()  
*Constroi uma Lista.*
- [Pessoa \\*](#) [CriaPessoa](#) (char \*nome, int nc)  
*Cria um nodo [Pessoa](#).*
- [ListaPessoa \\*](#) [CriaNodoListaPessoas](#) ([Pessoa \\*](#)c)  
*Cria novo nodo para a Lista de Pessoas Copia para o nodo da lista a informação de uma pessoa.*
- [ListaPessoa \\*](#) [InserePessoaListaPessoas](#) ([ListaPessoa \\*](#)h, [Pessoa \\*](#)p)  
*Insere pessoa na Lista de Pessoas.*
- [ListaPessoa \\*](#) [ProcuraPessoa](#) ([ListaPessoa \\*](#)inicio, int nc)
- void [MostraTodasPessoas](#) ([ListaPessoa \\*](#)h)
- [ListaPessoa \\*](#) [GetAllPessoas](#) (char \*fileName)  
*Carrega informação sobre Pessoas.*
- [CONTROLERROR](#) [SavePessoas](#) ([ListaPessoa \\*](#)h, char \*fileName)  
*Preserva informação da [Pessoa](#).*
- void [MostraContactosPessoa](#) ([ListaPessoa \\*](#)inicio, int nc)
- [ListaPessoa \\*](#) [InsereContactoPessoa](#) ([ListaPessoa \\*](#)h, Cont acto \*c, int nc)
- bool [SaveAll](#) ([ListaPessoa \\*](#)h, char \*fileName)  
*Pessoas e respectivos contactos.*
- [ListaPessoa \\*](#) [GetAllContacts](#) (char \*fileName, [ListaPessoa \\*](#)h)
- [ListaPessoa \\*](#) [GetData](#) (char \*fileName)  
*Carrega dados de entrada.*

### 5.7.1 Detailed Description

Lista Ligadas Simples (versão 1)

Author

lufer

Date

2022

Dados globais para uma Lista Ligada Simples de Pessoas

**Bug** No known bugs.

Definition in file [Pessoa.h](#).

### 5.7.2 Macro Definition Documentation

#### 5.7.2.1 M

```
#define M 20
```

Definition at line 16 of file [Pessoa.h](#).

### 5.7.3 Typedef Documentation

#### 5.7.3.1 CONTROLERROR

```
typedef enum CONTROLERROR CONTROLERROR
```

#### 5.7.3.2 ListaPessoa

```
typedef struct ListaPessoa ListaPessoa
```

Lista de Pessoas Informação sobre a [Pessoa](#), os seus contactos e apontador para outra [Pessoa](#).

### 5.7.3.3 Pessoa

```
typedef struct Pessoa Pessoa
```

Informação de uma pessoa.

### 5.7.3.4 TodaInformacaoPessoa

```
typedef struct TodaInformacaoPessoa TodaInformacaoPessoa
```

Preservar dados em ficheiro

## 5.7.4 Enumeration Type Documentation

### 5.7.4.1 CONTROLERROR

```
enum CONTROLERROR
```

Enumerator

NO_ERROR	
NO_FILE	
PRESIDENT	
NO_DATA	

Definition at line 37 of file [Pessoa.h](#).

## 5.7.5 Function Documentation

### 5.7.5.1 CriaLista()

```
Pessoa * CriaLista ( )
```

Constroi uma Lista.

Returns

Lista Vazia

Definition at line 19 of file [Pessoas.c](#).

### 5.7.5.2 CriaNodoListaPessoas()

```
ListaPessoa * CriaNodoListaPessoas (
    Pessoa * c )
```

Cria novo nodo para a Lista de Pessoas Copia para o nodo da lista a informação de uma pessoa.

#### Parameters

in	<i>c</i>	Novo contacto
out	<i>Apontador</i>	para nodo criado

Definition at line 43 of file [Pessoas.c](#).

### 5.7.5.3 CriaPessoa()

```
Pessoa * CriaPessoa (
    char * nome,
    int nc )
```

Cria um nodo [Pessoa](#).

#### Parameters

<i>nome</i>	Nome da <a href="#">Pessoa</a>
<i>nc</i>	Número de Contribuinte

#### Returns

[Pessoa](#)

Definition at line 30 of file [Pessoas.c](#).

### 5.7.5.4 GetAllContacts()

```
ListaPessoa * GetAllContacts (
    char * fileName,
    ListaPessoa * h )
```

Carrega todos os contactos para cada [Pessoa](#)

Definition at line 206 of file [Pessoas.c](#).

### 5.7.5.5 GetAllPessoas()

```
ListaPessoa * GetAllPessoas (
    char * fileName )
```

Carrega informação sobre Pessoas.

**Parameters**

<i>fileName</i>	
-----------------	--

**Returns**

Lista de Pessoas

feof(fp)

Definition at line 126 of file [Pessoas.c](#).

**5.7.5.6 GetData()**

```
ListaPessoa * GetData (
    char * fileName )
```

Carrega dados de entrada.

**Parameters**

<i>fileName</i>	
-----------------	--

**Returns**

Definition at line 234 of file [Pessoas.c](#).

**5.7.5.7 InsereContactoPessoa()**

```
ListaPessoa * InsereContactoPessoa (
    ListaPessoa * h,
    Cont acto * c,
    int nc )
```

**5.7.5.8 InserePessoaListaPessoas()**

```
ListaPessoa * InserePessoaListaPessoas (
    ListaPessoa * h,
    Pessoa * p )
```

Insere pessoa na Lista de Pessoas.

## Parameters

in	<i>p</i>	Nova pessoa
in	<i>Apontador</i>	para inicio da lista
out	<i>Apontador</i>	para inicio da lista

Definition at line 58 of file [Pessoas.c](#).

### 5.7.5.9 MostraContactosPessoa()

```
void MostraContactosPessoa (
    ListaPessoa * inicio,
    int nc )
```

Definition at line 106 of file [Pessoas.c](#).

### 5.7.5.10 MostraTodasPessoas()

```
void MostraTodasPessoas (
    ListaPessoa * h )
```

Definition at line 96 of file [Pessoas.c](#).

### 5.7.5.11 ProcuraPessoa()

```
ListaPessoa * ProcuraPessoa (
    ListaPessoa * inicio,
    int nc )
```

Definition at line 72 of file [Pessoas.c](#).

### 5.7.5.12 SaveAll()

```
bool SaveAll (
    ListaPessoa * h,
    char * fileName )
```

Pessoas e respectivos contactos.

**Parameters**

<i>h</i>	
<i>fileName</i>	

**Returns**

Definition at line 175 of file [Pessoas.c](#).

**5.7.5.13 SavePessoas()**

```
CONTROLERROR SavePessoas (
    ListaPessoa * h,
    char * fileName )
```

Preserva informação da [Pessoa](#).

**Parameters**

<i>h</i>	
<i>fileName</i>	

**Returns**

Definition at line 152 of file [Pessoas.c](#).

**5.8 Pessoa.h**

[Go to the documentation of this file.](#)

```
00001
00010 #pragma once
00011 #pragma warning( disable : 4996 ) //evita MSG ERROS: _CRT_SECURE_NO_WARNINGS
00012 #include <stdio.h>
00013 #include "Contatos.h"
00014 #include <stdbool.h>
00015
00016 #define M 20
00017
00018 #pragma region InformaçãoMemória
00022 typedef struct Pessoa {
00023     int nc;
00024     char nome[M];
00025 }Pessoa;
00026
00031 typedef struct ListaPessoa {
00032     struct Pessoa fichaPessoa;
00033     struct ListaContactos* listaContactos;
00034     struct ListaPessoa* proxPessoa;
00035 }ListaPessoa;
00036
00037 typedef enum CONTROLERROR {
```



```

00038     NO_ERROR = 1,
00039     NO_FILE =45,
00040     PRESIDENT=100,
00041     NO_DATA=2
00042 }CONTROLEERROR;
00043
00044 #pragma region GerePessoa
00045
00046 Pessoa* CriaLista();
00047 Pessoa* CriaPessoa(char* nome, int nc);
00048 ListaPessoa* CriaNodoListaPessoas(Pessoa* c);
00049 ListaPessoa* InserePessoaListaPessoas(ListaPessoa* h, Pessoa* p);
00050 ListaPessoa* ProcuraPessoa(ListaPessoa* inicio, int nc);
00051 void MostraTodasPessoas(ListaPessoa* h);
00052 ListaPessoa* GetAllPessoas(char* fileName);
00053
00054 CONTROLEERROR SavePessoas(ListaPessoa* h, char* fileName);
00055
00056 #pragma endregion
00057
00058 #pragma region GereListaContactos
00059
00060 void MostraContactosPessoa(ListaPessoa* inicio, int nc);
00061 ListaPessoa* InsereContactoPessoa(ListaPessoa* h, Cont
00062     acto* c, int nc);
00063
00064 #pragma endregion
00065
00066 #pragma endregion
00067
00068 #pragma region InformaçãoFicheiro
00069
00073 typedef struct TodaInformacaoPessoa {
00074     int nc;
00075     Contacto contacto;
00076 }TodaInformacaoPessoa;
00077
00078 bool SaveAll(ListaPessoa* h, char* fileName);
00079
00080 ListaPessoa* GetAllContacts(char* fileName, ListaPessoa* h);
00081 ListaPessoa* GetData(char* fileName);
00082
00083 #pragma endregion

```

## 5.9 D:/Temp/PROG-EGI/Aulas/GereContactos/Pessoas.c File Reference

Lista Ligadas Simples (versão 1)

```

#include "Pessoa.h"
#include "Contatos.h"

```

### Macros

- #define **MAX** 100

### Functions

- **Pessoa** \* **CriaLista** ()  
*Constroi uma Lista.*
- **Pessoa** \* **CriaPessoa** (char \*nome, int nc)  
*Cria um nodo **Pessoa**.*
- **ListaPessoa** \* **CriaNodoListaPessoas** (**Pessoa** \*c)  
*Cria novo nodo para a Lista de Pessoas Copia para o nodo da lista a informação de uma pessoa.*
- **ListaPessoa** \* **InserePessoaListaPessoas** (**ListaPessoa** \*h, **Pessoa** \*p)  
*Insere pessoa na Lista de Pessoas.*

- [ListaPessoa](#) \* [ProcuraPessoa](#) ([ListaPessoa](#) \*inicio, int nc)
- [ListaPessoa](#) \* [InsereContactoPessoa](#) ([ListaPessoa](#) \*h, [Contacto](#) \*c, int nc)
- void [MostraTodasPessoas](#) ([ListaPessoa](#) \*h)
- void [MostraContactosPessoa](#) ([ListaPessoa](#) \*inicio, int nc)
- [ListaPessoa](#) \* [GetAllPessoas](#) (char \*fileName)  
*Carrega informação sobre Pessoas.*
- bool [SavePessoas](#) ([ListaPessoa](#) \*h, char \*fileName)  
*Preserva informação da [Pessoa](#).*
- bool [SaveAll](#) ([ListaPessoa](#) \*h, char \*fileName)  
*Pessoas e respectivos contactos.*
- [ListaPessoa](#) \* [GetAllContacts](#) (char \*fileName, [ListaPessoa](#) \*h)
- [ListaPessoa](#) \* [GetData](#) (char \*fileName)  
*Carrega dados de entrada.*

## 5.9.1 Detailed Description

Lista Ligadas Simples (versão 1)

Author

lufer

Date

2022

Metodos para manipular uma Lista Ligada Simples de Pessoas

**Bug** No known bugs.

Definition in file [Pessoas.c](#).

## 5.9.2 Macro Definition Documentation

### 5.9.2.1 MAX

```
#define MAX 100
```

Definition at line 226 of file [Pessoas.c](#).

## 5.9.3 Function Documentation

### 5.9.3.1 CriaLista()

```
Pessoa * CriaLista ( )
```

Constroi uma Lista.

#### Returns

Lista Vazia

Definition at line 19 of file [Pessoas.c](#).

### 5.9.3.2 CriaNodoListaPessoas()

```
ListaPessoa * CriaNodoListaPessoas (
    Pessoa * c )
```

Cria novo nodo para a Lista de Pessoas Copia para o nodo da lista a informação de uma pessoa.

#### Parameters

in	<i>c</i>	Novo contacto
out	<i>Apontador</i>	para nodo criado

Definition at line 43 of file [Pessoas.c](#).

### 5.9.3.3 CriaPessoa()

```
Pessoa * CriaPessoa (
    char * nome,
    int nc )
```

Cria um nodo [Pessoa](#).

#### Parameters

<i>nome</i>	Nome da <a href="#">Pessoa</a>
<i>nc</i>	Número de Contribuinte

#### Returns

[Pessoa](#)

Definition at line 30 of file [Pessoas.c](#).

#### 5.9.3.4 GetAllContacts()

```
ListaPessoa * GetAllContacts (
    char * fileName,
    ListaPessoa * h )
```

Carrega todos os contactos para cada Pessoa

Definition at line 206 of file [Pessoas.c](#).

#### 5.9.3.5 GetAllPessoas()

```
ListaPessoa * GetAllPessoas (
    char * fileName )
```

Carrega informação sobre Pessoas.

##### Parameters

<i>fileName</i>	
-----------------	--

##### Returns

Lista de Pessoas

feof(fp)

Definition at line 126 of file [Pessoas.c](#).

#### 5.9.3.6 GetData()

```
ListaPessoa * GetData (
    char * fileName )
```

Carrega dados de entrada.

##### Parameters

<i>fileName</i>	
-----------------	--

##### Returns

Definition at line 234 of file [Pessoas.c](#).

### 5.9.3.7 InsereContactoPessoa()

```
ListaPessoa * InsereContactoPessoa (
    ListaPessoa * h,
    Contacto * c,
    int nc )
```

Definition at line 83 of file [Pessoas.c](#).

### 5.9.3.8 InserePessoaListaPessoas()

```
ListaPessoa * InserePessoaListaPessoas (
    ListaPessoa * h,
    Pessoa * p )
```

Insere pessoa na Lista de Pessoas.

#### Parameters

in	<i>p</i>	Nova pessoa
in	<i>Apontador</i>	para inicio da lista
out	<i>Apontador</i>	para inicio da lista

Definition at line 58 of file [Pessoas.c](#).

### 5.9.3.9 MostraContactosPessoa()

```
void MostraContactosPessoa (
    ListaPessoa * inicio,
    int nc )
```

Definition at line 106 of file [Pessoas.c](#).

### 5.9.3.10 MostraTodasPessoas()

```
void MostraTodasPessoas (
    ListaPessoa * h )
```

Definition at line 96 of file [Pessoas.c](#).

#### 5.9.3.11 ProcuraPessoa()

```
ListaPessoa * ProcuraPessoa (
    ListaPessoa * inicio,
    int nc )
```

Definition at line 72 of file [Pessoas.c](#).

#### 5.9.3.12 SaveAll()

```
bool SaveAll (
    ListaPessoa * h,
    char * fileName )
```

Pessoas e respectivos contactos.

##### Parameters

<i>h</i>	
<i>fileName</i>	

##### Returns

Definition at line 175 of file [Pessoas.c](#).

#### 5.9.3.13 SavePessoas()

```
bool SavePessoas (
    ListaPessoa * h,
    char * fileName )
```

Preserva informação da [Pessoa](#).

##### Parameters

<i>h</i>	
<i>fileName</i>	

##### Returns

Definition at line 152 of file [Pessoas.c](#).

## 5.10 Pessoas.c

[Go to the documentation of this file.](#)

```

00001
00010 #include "Pessoa.h"
00011 #include "Contatos.h"
00012
00013
00019 Pessoa* CriaLista() {
00020     return NULL;
00021 }
00022
00030 Pessoa* CriaPessoa(char* nome, int nc) {
00031     Pessoa* aux = (Pessoa*)calloc(1, sizeof(Pessoa));
00032     aux->nc = nc;
00033     strcpy(aux->nome, nome);
00034     return aux;
00035 }
00036
00043 ListaPessoa* CriaNodoListaPessoas(Pessoa* c) {
00044     ListaPessoa* nova = (ListaPessoa*)calloc(1, sizeof(ListaPessoa));
00045     nova->fichaPessoa.nc=c->nc;
00046     strcpy(nova->fichaPessoa.nome, c->nome);
00047     nova->listaContactos = NULL; // no início não tem contactos
00048     nova->proxPessoa = NULL;
00049     return nova;
00050 }
00051
00058 ListaPessoa* InserePessoaListaPessoas(ListaPessoa* h, Pessoa* p) {
00059     if (p == NULL) return h; //se nova não tem dados
00060     //Cria novo nodo da lista de pessoas
00061     ListaPessoa* nova = CriaNodoListaPessoas(p);
00062     if (h == NULL) h = nova; //se lista é vazia
00063     else { // insere ordenado pelo nc
00064         nova->proxPessoa = h;
00065         h = nova;
00066     }
00067     return h;
00068 }
00069
00072 ListaPessoa* ProcuraPessoa(ListaPessoa* inicio, int nc) {
00073     ListaPessoa* aux = inicio;
00074     while (aux) {
00075         if (aux->fichaPessoa.nc == nc) return aux; //se encontrou
00076         aux = aux->proxPessoa;
00077     }
00078     return NULL; //se não encontrou
00079 }
00080
00083 ListaPessoa* InsereContactoPessoa(ListaPessoa* h, Contacto* c, int nc) {
00084     if (h == NULL) return NULL; // se lista vazia
00085     if (c == NULL) return h; //se contacto não tem informação
00086
00087     ListaPessoa* aux = ProcuraPessoa(h, nc);
00088     if (aux) { //se existe essa pessoa
00089         aux->listaContactos = InsereContactoListaContactos(aux->listaContactos, c);
00090     }
00091     return h;
00092 }
00093
00096 void MostraTodasPessoas(ListaPessoa* h) {
00097     ListaPessoa* aux = h;
00098     while (aux) {
00099         printf("Pessoa: NC=%d - Nome= %s\n", aux->fichaPessoa.nc, aux->fichaPessoa.nome);
00100         aux = aux->proxPessoa;
00101     }
00102 }
00103
00106 void MostraContactosPessoa(ListaPessoa* inicio, int nc) {
00107     ListaPessoa* aux = inicio;
00108     aux = ProcuraPessoa(inicio, nc);
00109     if (aux) {
00110         ListaContactos* inicioContactos = aux->listaContactos;
00111         printf("Pessoa: %d\n", aux->fichaPessoa.nc);
00112         MostraContactos(inicioContactos);
00113     }
00114 }
00115
00116
00117
00118 #pragma region GereDataFile
00119
00126 ListaPessoa* GetAllPessoas(char* fileName) {
00127     FILE* fp;
00128     ListaPessoa* h = NULL;

```

```

00129     ListaPessoa* aux;
00130     Pessoa p;
00131
00132     if ((fp = fopen(fileName, "rb")) == NULL)
00133         return NULL;
00134     while (fread(&p, sizeof(p), 1, fp)) {
00135         if (ProcuraPessoa(h, p.nc) == NULL) { //se pessoa ainda não está em memória
00136             aux = CriaNodoListaPessoas(&p);
00137             h = InserePessoaListaPessoas(h, aux);
00138             continue;
00139         }
00140     }
00141     fclose(fp);
00142     return h;
00143 }
00144
00152 bool SavePessoas(ListaPessoa* h, char* fileName) {
00153     if (h == NULL) return false;
00154     FILE* fp;
00155     if ((fp = fopen(fileName, "wb")) == NULL)
00156         return false;
00157     ListaPessoa* aux = h;
00158     while (aux) {
00159         fwrite(&aux->fichaPessoa, sizeof(Pessoa), 1, fp);
00160         aux = aux->proxPessoa;
00161     }
00162     fclose(fp);
00163     return true;
00164 }
00165
00175 bool SaveAll(ListaPessoa* h, char* fileName) {
00176     if (h == NULL)
00177         return false;
00178     FILE* fp;
00179
00180     if ((fp = fopen(fileName, "wb")) == NULL)
00181         return false;
00182
00183     //grava n registos no ficheiro
00184     ListaPessoa* aux = h;
00185     TodaInformacaoPessoa auxFile; //struct para gravar em ficheiro!
00186     while (aux) {
00187         //Colocar no registo de ficheiro a inf que está no registo de memória
00188         auxFile.nc = aux->fichaPessoa.nc;
00189         //percore a lista de contactos
00190         ListaContactos* auxContatos = aux->listaContactos;
00191         while (auxContatos) {
00192             strcpy(auxFile.contacto.desc, auxContatos->contacto.desc);
00193             strcpy(auxFile.contacto.valor, auxContatos->contacto.valor);
00194             fwrite(&auxFile, sizeof(auxFile), 1, fp);
00195             auxContatos = auxContatos->proxContacto;
00196         }
00197         aux = aux->proxPessoa;
00198     }
00199     fclose(fp);
00200     return true;
00201 }
00202
00206 ListaPessoa* GetAllContacts(char* fileName, ListaPessoa* h) {
00207     FILE* fp;
00208     ListaPessoa* aux = h;
00209     Pessoa p;
00210
00211     if (h == NULL) return NULL;
00212
00213     if ((fp = fopen(fileName, "rb")) == NULL)
00214         return NULL;
00215
00216     TodaInformacaoPessoa auxFile;
00217     while (fread(&auxFile, sizeof(auxFile), 1, fp)) {
00218         //aux = ProcuraPessoa(h, auxFile.nc);
00219         //ListaPessoa* InsereContactoPessoa(ListaPessoa* h, Contacto* c, int nc)
00220         h = InsereContactoPessoa(h, &auxFile.contacto, auxFile.nc);
00221     }
00222     fclose(fp);
00223     return h;
00224 }
00225
00226 #define MAX 100
00227
00234 ListaPessoa* GetData(char* fileName) {
00235     FILE* fp;
00236     ListaPessoa* h = NULL;
00237     ListaPessoa* aux; //auxiliar
00238     Pessoa p; //auxiliar

```



```
00239     Contacto c;           //auxiliar
00240     char linhaFicheiro[MAX];
00241
00242     if ((fp = fopen(fileName, "r")) == NULL) return NULL;
00243
00244     while (fgets(linhaFicheiro,MAX,fp) != NULL)
00245     {
00246         //[^0-9]
00247         //[^a-zA-Z]
00248         sscanf(linhaFicheiro,"%[^;];%d;%[^;];%s", p.nome, &p.nc, c.desc, c.valor);
00249
00250         if (ProcuraPessoa(h, p.nc) == NULL) { //se pessoa ainda não está em memória
00251             //Insere Pessoa
00252             aux = CriaNodoListaPessoas(&p);
00253             h = InserePessoaListaPessoas(h, aux);
00254         }
00255         //Insere contacto
00256         h = InsereContactoPessoa(h, &c, p.nc);
00257     }
00258     fclose(fp);
00259     return h;
00260 }
00261 #pragma endregion
00262
00263
```



# Index

- Contacto, [7](#)
  - Contatos.h, [16](#)
  - desc, [7](#)
  - valor, [7](#)
- contacto
  - ListaContactos, [8](#)
  - TodaInformacaoPessoa, [11](#)
- Contactos.c
  - CriaContacto, [14](#)
  - CriaNodoListaContactos, [14](#)
  - InsererContactoListaContactos, [14](#)
  - MostraContactos, [14](#)
- Contatos.h
  - Contacto, [16](#)
  - CriaContacto, [17](#)
  - CriaNodoListaContactos, [17](#)
  - InsererContactoListaContactos, [17](#)
  - ListaContactos, [16](#)
  - M, [16](#)
  - MostraContactos, [18](#)
  - N, [16](#)
- CONTROLERROR
  - Pessoa.h, [21](#), [22](#)
- CriaContacto
  - Contactos.c, [14](#)
  - Contatos.h, [17](#)
- CriaLista
  - Pessoa.h, [22](#)
  - Pessoas.c, [28](#)
- CriaNodoListaContactos
  - Contactos.c, [14](#)
  - Contatos.h, [17](#)
- CriaNodoListaPessoas
  - Pessoa.h, [22](#)
  - Pessoas.c, [29](#)
- CriaPessoa
  - Pessoa.h, [23](#)
  - Pessoas.c, [29](#)
- D:/Temp/PROG-EGI/Aulas/GereContactos/Contactos.c, [13](#), [15](#)
- D:/Temp/PROG-EGI/Aulas/GereContactos/Contatos.h, [15](#), [18](#)
- D:/Temp/PROG-EGI/Aulas/GereContactos/Main.c, [18](#), [19](#)
- D:/Temp/PROG-EGI/Aulas/GereContactos/Pessoa.h, [19](#), [26](#)
- D:/Temp/PROG-EGI/Aulas/GereContactos/Pessoas.c, [27](#), [33](#)
- desc
  - Contacto, [7](#)
- fichaPessoa
  - ListaPessoa, [9](#)
- GetAllContacts
  - Pessoa.h, [23](#)
  - Pessoas.c, [29](#)
- GetAllPessoas
  - Pessoa.h, [23](#)
  - Pessoas.c, [30](#)
- GetData
  - Pessoa.h, [24](#)
  - Pessoas.c, [30](#)
- InsererContactoListaContactos
  - Contactos.c, [14](#)
  - Contatos.h, [17](#)
- InsererContactoPessoa
  - Pessoa.h, [24](#)
  - Pessoas.c, [30](#)
- InsererPessoaListaPessoas
  - Pessoa.h, [24](#)
  - Pessoas.c, [31](#)
- ListaContactos, [8](#)
  - contacto, [8](#)
  - Contatos.h, [16](#)
  - proxContacto, [8](#)
- listaContactos
  - ListaPessoa, [9](#)
- ListaPessoa, [9](#)
  - fichaPessoa, [9](#)
  - listaContactos, [9](#)
  - Pessoa.h, [21](#)
  - proxPessoa, [9](#)
- M
  - Contatos.h, [16](#)
  - Pessoa.h, [21](#)
- main
  - Main.c, [18](#)
- Main.c
  - main, [18](#)
- MAX
  - Pessoas.c, [28](#)
- MostraContactos
  - Contactos.c, [14](#)
  - Contatos.h, [18](#)
- MostraContactosPessoa
  - Pessoa.h, [25](#)

- Pessoas.c, 31
- MostraTodasPessoas
  - Pessoa.h, 25
  - Pessoas.c, 31
- N
  - Contatos.h, 16
- nc
  - Pessoa, 10
  - TodaInformacaoPessoa, 11
- NO\_DATA
  - Pessoa.h, 22
- NO\_ERROR
  - Pessoa.h, 22
- NO\_FILE
  - Pessoa.h, 22
- nome
  - Pessoa, 10
- Pessoa, 10
  - nc, 10
  - nome, 10
  - Pessoa.h, 21
- Pessoa.h
  - CONTROLERROR, 21, 22
  - CriaLista, 22
  - CriaNodoListaPessoas, 22
  - CriaPessoa, 23
  - GetAllContacts, 23
  - GetAllPessoas, 23
  - GetData, 24
  - InsererContactoPessoa, 24
  - InsererPessoaListaPessoas, 24
  - ListaPessoa, 21
  - M, 21
  - MostraContactosPessoa, 25
  - MostraTodasPessoas, 25
  - NO\_DATA, 22
  - NO\_ERROR, 22
  - NO\_FILE, 22
  - Pessoa, 21
  - PRESIDENT, 22
  - ProcuraPessoa, 25
  - SaveAll, 25
  - SavePessoas, 26
  - TodaInformacaoPessoa, 22
- Pessoas.c
  - CriaLista, 28
  - CriaNodoListaPessoas, 29
  - CriaPessoa, 29
  - GetAllContacts, 29
  - GetAllPessoas, 30
  - GetData, 30
  - InsererContactoPessoa, 30
  - InsererPessoaListaPessoas, 31
  - MAX, 28
  - MostraContactosPessoa, 31
  - MostraTodasPessoas, 31
  - ProcuraPessoa, 31
  - SaveAll, 32
  - SavePessoas, 32
- PRESIDENT
  - Pessoa.h, 22
- ProcuraPessoa
  - Pessoa.h, 25
  - Pessoas.c, 31
- proxContacto
  - ListaContactos, 8
- proxPessoa
  - ListaPessoa, 9
- SaveAll
  - Pessoa.h, 25
  - Pessoas.c, 32
- SavePessoas
  - Pessoa.h, 26
  - Pessoas.c, 32
- TodaInformacaoPessoa, 11
  - contacto, 11
  - nc, 11
  - Pessoa.h, 22
- valor
  - Contacto, 7