Projeto Final Labs

Gerado por Doxygen 1.9.1

Chapter 1

Página principal

1.1 Trabalho Prático

Licenciatura em Engenharia de Sistemas Informáticos 2023-24

Laboratórios de Informática

1.1.1 grupo 35

Número	Nome	Tarefas
22999	Joana Pimenta	Criação dos menus, make file e relatório em latex
23000	Diogo Marques	Criação da lógica para os pontos pedidos no relatório e documentação em Doxygen
23135	Miguel Simões	Criação do sistema para ler os arquivos, criação das listas e realização dos testes

1.1.2 organização

`doc/` documentação com o relatório

`src/` Código da solução desenvolvida

`documentação doxygen pdf` Documentação gerada pelo doxygen

`relatório` Relatório do trabalho

1.1.3 como compilar

```
cd src
make all
```

1.1.4 como executar

```
./prog_nutricional
Parâmetros (Todas as parâmetros são opcionais, o programa funciona sem nenhum):
    -ajuda -> mostra o menu de ajuda do programa
    -bin   -> ler arquivos em binário
    -tab   -> ler arquivos separados por tabuladores
    O -bin e o -tab podem ser usados em simultâneo, já o -ajuda termina o programa ao fim de dar display ao
       menu de ajuda
```

Chapter 2

Índice dos componentes

2.1 Lista de componentes

Lista de classes, estruturas, uniões e interfaces com uma breve descrição:

Chapter 3

Índice dos ficheiros

3.1 Lista de ficheiros

Lista de todos os ficheiros documentados com uma breve descrição:

Chapter 4

Documentação da classe

4.1 Referência à estrutura Cliente

Atributos Públicos

- int **pacienteID**
- char **nome** [30]
- int **telefone**

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

- /root/d-35/src/structs.h

4.2 Referência à estrutura Data

Atributos Públicos

- int **dia**
- int **mes**
- int **ano**

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

- /root/d-35/src/structs.h

4.3 Referência à estrutura DietaRealizada

Atributos Públicos

- int **pacienteID**
- Data **data**
- int **refeicao**
- char **alimento** [50]
- int **calorias**

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

- /root/d-35/src/structs.h

4.4 Referência à estrutura elementoCliente

Atributos Públicos

- Cliente **cliente**
- struct elementoCliente * **prox**

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

- /root/d-35/src/structs.h

4.5 Referência à estrutura elementoDietaRelizada

Atributos Públicos

- DietaRealizada **dietaRealizada**
- struct elementoDietaRelizada * **prox**

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

- /root/d-35/src/structs.h

4.6 Referência à estrutura elementoPlanoNutricional

Atributos Públicos

- PlanoNutricional **planoNutricional**
- struct elementoPlanoNutricional * **prox**

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

- /root/d-35/src/structs.h

4.7 Referência à estrutura elementoRefeicoesPlaneadas

Atributos Públicos

- RefeicoesPlaneadas **refeicoesPlaneadas**
- struct elementoRefeicoesPlaneadas * **prox**

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

- /root/d-35/src/structs.h

4.8 Referência à estrutura PlanoNutricional

Atributos Públicos

- int **pacienteID**
- Data **dataInicio**
- Data **dataFim**
- int **refeicao**
- int **min_cal**
- int **max_cal**

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

- /root/d-35/src/structs.h

4.9 Referência à estrutura RefeicoesPlaneadas

Atributos Públicos

- int **pacienteID**
- char **nome** [30]
- int **refeicao**
- Data **dataInicio**
- Data **dataFim**
- int **min_cal**
- int **max_cal**
- int **consumo_cal**
- int **n_compridas**

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

- /root/d-35/src/structs.h

Chapter 5

Documentação do ficheiro

5.1 Referência ao ficheiro /root/d-35/src/functions.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "functions.h"
```

Funções

- bool verificarParametros (int argc, char ∗argv[ ])

  Função para verificar os parâmetros passados pelo utilizador.
- void lerInteiro (int ∗n)

  Função para ler um número inteiro.
- ListaCliente ∗ criarListasClientes ()

  Função para criar a lista de clientes.
- ListaDietaRealizada ∗ criarListasDietasRealizadas ()

  Função para criar a lista de dietas realizadas.
- ListaPlanoNutricional ∗ criarListasPlanosNutricionais ()

  Função para criar a lista de planos nutricionais.
- ListaRefeicoesPlaneadas ∗ criarListasRefeicoesPlaneadas ()

  Função para criar a lista de refeições planeadas.
- bool criarListas ()

  Função para criar as listas na inicialização do programa.
- void liberarListaClientes (ListaCliente ∗li)

  Função para liberar a lista de clientes.
- void liberarListaDietasRealizadas (ListaDietaRealizada ∗li)

  Função para liberar a lista de dietas realizadas.
- void liberarListaPlanosNutricionais (ListaPlanoNutricional ∗li)

  Função para liberar a lista de planos nutricionais.
- void liberarListaRefeicoesPlaneadas (ListaRefeicoesPlaneadas ∗li)

  Função para liberar a lista de refeições planeadas.
- void endProgram ()

  Função para terminar o programa e apagar as listas.

- bool checkClienteID (ListaCliente ∗li, int id)

  Função para verificar se o ID do cliente existe.
- bool checkPlanoNutricionalByClienteDataRef (ListaPlanoNutricional ∗li, int idCliente, Data data, int refeicao)

  Função para verificar se já existe um plano nutricional para o cliente dentro do intervalo de tempo.
- bool insertCliente (ListaCliente ∗li, Cliente cliente)

  Função para inserir um novo cliente na lista.
- bool insertDietaRealizada (ListaDietaRealizada ∗li, DietaRealizada dietaRealizada)

  Função para inserir uma dieta realizada.
- bool insertPlanoNutricional (ListaPlanoNutricional ∗li, PlanoNutricional planoNutricional)

  Função para inserir um plano nutricional.
- bool insertRefeicaoPlaneada (ListaRefeicoesPlaneadas ∗li, RefeicoesPlaneadas refeicoesPlaneadas)

  Função para inserir uma refeição planeada.
- int convertRefeicao (char ∗refeicao)

  Função para converter a string de uma refeição para número.
- char ∗ convertRefeicaoString (int refeicao)

  Função para converter o número da refeição para a sua string.
- bool fileExists (char ∗fileName)

  Função para verificar se o arquivo existe.
- bool carregarClientes ()

  Função para carregar os clientes.
- bool carregarDietasRealizadas ()

  Função para carregar as dietas realizadas.
- bool carregarPlanosNutricionais ()

  Função para carregar os planos nutricionais.
- void displayClientes ()

  Função para fazer display dos clientes inseridos na lista.
- void displayDietasRealizadas ()

  Função para fazer display das dietas realizadas inseridas na lista.
- void displayPlanosNutricionais ()

  Função para fazer display dos planos nutricionais inseridos na lista.
- bool isAnoBissexto (int ano)

  Função para verificar se um ano é bissexto.
- int diaMes (int mes, int ano)

  Função para obter os dias do mês.
- void ordenarClientesDecrescente (ListaCliente ∗li)

  Função para ordenar os clientes de forma decrescente.
- bool isValidDataRange (Data dataInicio, Data dataFim)

  Função para verificar se as datas fornecidas formam um range válido.
- bool isInDataRange (Data data, Data dataInicio, Data dataFim)

  Função para verificar se a data está dentro do intervalo de datas.
- bool iSDateRangeInsideOther (Data dataInicio, Data dataFim, Data dataInicio2, Data dataFim2)

  Função para verificar se um range de datas está dentro do outro.
- Data perguntarData ()

  Função para perguntar a data ao utilizador.
- void clientesPassaramCaloriasPeriodoTempo ()

  Função para o ponto 2 do trabalho.
- void clientesPassaramPlanoNutricionalPeriodoTempoDecrescente ()

  Função para o ponto 3 do trabalho.
- void listarPlanoNutricionalPacienteRefeicaoPeriodoTempo ()

  Função para o ponto 4 do trabalho.
- void calcularMediaCalRefeicaoPassadoPeriodoTempo ()

  Função para o ponto 5 do trabalho.
- void gerarTabelaRefeicoesPlaneadas ()

  Função para o ponto 6 do trabalho.

Variáveis

- bool **arquivoTab** = false
- bool **arquivoBin** = false
- ListaCliente * **listaCliente**
- ListaDietaRealizada * **listaDietaRealizada**
- ListaPlanoNutricional * **listaPlanoNutricional**
- ListaRefeicoesPlaneadas * **listaRefeicoesPlaneadas**

5.1.1 Descrição detalhada

Versão

0.1

Data

2023-12-21

Copyright

Copyright (c) 2023

5.1.2 Documentação das funções

5.1.2.1 carregarClientes()

```
bool carregarClientes ( )
```

Função para carregar os clientes.

Retorna

true * Se os clientes foram carregados com sucesso

false * Se os clientes não foram carregados com sucesso

5.1.2.2 carregarDietasRealizadas()

```
bool carregarDietasRealizadas ( )
```

Função para carregar as dietas realizadas.

Retorna

true * Se as dietas realizadas foram carregadas com sucesso

false * Se as dietas realizadas não foram carregadas com sucesso

5.1.2.3 carregarPlanosNutricionais()

```
bool carregarPlanosNutricionais ( )
```

Função para carregar os planos nutricionais.

Retorna

true ∗ Se os planos nutricionais foram carregados com sucesso

false ∗ Se os planos nutricionais não foram carregados com sucesso

5.1.2.4 checkClienteID()

```
bool checkClienteID (
            ListaCliente * li,
            int id )
```

Função para verificar se o ID do cliente existe.

Parâmetros

li	∗ Lista de clientes
id	∗ ID do cliente

Retorna

true ∗ Se o ID do cliente existe

false ∗ Se o ID do cliente não existe

5.1.2.5 checkPlanoNutricionalByClienteDataRef()

```
bool checkPlanoNutricionalByClienteDataRef (
            ListaPlanoNutricional * li,
            int idCliente,
            Data data,
            int refeicao )
```

Função para verificar se já existe um plano nutricional para o cliente dentro do intervalo de tempo.

Parâmetros

li	∗ Lista de planos nutricionais
idCliente	∗ ID do cliente
data	∗ Data
refeicao	∗ Refeição

Retorna

true ∗ Se já existe um plano nutricional para o cliente dentro do intervalo de tempo

false ∗ Se não existe um plano nutricional para o cliente dentro do intervalo de tempo

5.1.2.6 convertRefeicao()

```
int convertRefeicao (
            char * refeicao )
```

Função para converter a string de uma refeição para número.

Parâmetros

refeicao	∗ String da refeição

Retorna

int ∗ Número da refeição

5.1.2.7 convertRefeicaoString()

```
char* convertRefeicaoString (
            int refeicao )
```

Função para converter o número da refeição para a sua string.

Parâmetros

refeicao	∗ Número da refeição

Retorna

char∗ ∗ String da refeição

5.1.2.8 criarListas()

```
bool criarListas ( )
```

Função para criar as listas na inicialização do programa.

Retorna

true ∗ Se as listas foram criadas

false ∗ Se as listas não foram criadas

5.1.2.9 criarListasClientes()

```
ListaCliente* criarListasClientes ( )
```

Função para criar a lista de clientes.

Retorna

ListaCliente** * Lista de clientes

5.1.2.10 criarListasDietasRealizadas()

```
ListaDietaRealizada* criarListasDietasRealizadas ( )
```

Função para criar a lista de dietas realizadas.

Retorna

ListaDietaRealizada** * Lista de dietas realizadas

5.1.2.11 criarListasPlanosNutricionais()

```
ListaPlanoNutricional* criarListasPlanosNutricionais ( )
```

Função para criar a lista de planos nutricionais.

Retorna

ListaPlanoNutricional** * Lista de planos nutricionais

5.1.2.12 criarListasRefeicoesPlaneadas()

```
ListaRefeicoesPlaneadas* criarListasRefeicoesPlaneadas ( )
```

Função para criar a lista de refeições planeadas.

Retorna

ListaRefeicoesPlaneadas** * Lista de refeições planeadas

5.1.2.13 diaMes()

```
int diaMes (
            int mes,
            int ano )
```

Função para obter os dias do mês.

Parâmetros

mes	∗ Mês
ano	∗ Ano

Retorna

int ∗ Número de dias do mês

5.1.2.14 fileExists()

```
bool fileExists (
            char * fileName )
```

Função para verificar se o arquivo existe.

Parâmetros

fileName	∗ Nome/Caminho do arquivo

Retorna

true ∗ Se o arquivo existir

false ∗ Se o arquivo não existir

5.1.2.15 insertCliente()

```
bool insertCliente (
            ListaCliente * li,
            Cliente cliente )
```

Função para inserir um novo cliente na lista.

Parâmetros

li	∗ Lista de clientes
cliente	∗ Dados do cliente

Retorna

true ∗ Se o cliente foi inserido com sucesso

false ∗ Se o cliente não foi inserido com sucesso

5.1.2.16 insertDietaRealizada()

```
bool insertDietaRealizada (
            ListaDietaRealizada * li,
            DietaRealizada dietaRealizada )
```

Função para inserir uma dieta realizada.

Parâmetros

li	∗ Lista de dietas realizadas
dietaRealizada	∗ Dados da dieta realizada

Retorna

true ∗ Se a dieta realizada foi inserida com sucesso

false ∗ Se a dieta realizada não foi inserida com sucesso

5.1.2.17 insertPlanoNutricional()

```
bool insertPlanoNutricional (
            ListaPlanoNutricional * li,
            PlanoNutricional planoNutricional )
```

Função para inserir um plano nutricional.

Parâmetros

li	∗ Lista de planos nutricionais
planoNutricional	∗ Dados do plano nutricional

Retorna

true ∗ Se o plano nutricional foi inserido com sucesso

false ∗ Se o plano nutricional não foi inserido com sucesso

5.1.2.18 insertRefeicaoPlaneada()

```
bool insertRefeicaoPlaneada (
            ListaRefeicoesPlaneadas * li,
            RefeicoesPlaneadas refeicoesPlaneadas )
```

Função para inserir uma refeição planeada.

Parâmetros

li	∗ Lista de refeições planeadas
refeicoesPlaneadas	∗ Dados da refeição planeada

Retorna

true ∗ Se a refeição planeada foi inserida com sucesso

false ∗ Se a refeição planeada não foi inserida com sucesso

5.1.2.19 isAnoBissexto()

```
bool isAnoBissexto (
            int ano )
```

Função para verificar se um ano é bissexto.

Parâmetros

ano	∗ Ano

Retorna

true ∗ Se o ano for bissexto

false ∗ Se o ano não for bissexto

5.1.2.20 iSDateRangeInsideOther()

```
bool iSDateRangeInsideOther (
            Data dataInicio,
            Data dataFim,
            Data dataInicio2,
            Data dataFim2 )
```

Função para verificar se um range de datas está dentro do outro.

Parâmetros

dataInicio	∗ Data de início do 1º range
dataFim	∗ Data de fim do 1º range
dataInicio2	∗ Data de início do 2º range
dataFim2	∗ Data de fim do 2º range

Retorna

true ∗ Se o 1º range estiver dentro do 2º range

false ∗ Se o 1º range não estiver dentro do 2º range

5.1.2.21 isInDataRange()

```
bool isInDataRange (
            Data data,
            Data dataInicio,
            Data dataFim )
```

Função para verificar se a data está dentro do intervalo de datas.

Parâmetros

data	∗ Data para verificar
dataInicio	∗ Data de início
dataFim	∗ Data de fim

Retorna

true ∗ Se a data estiver dentro do intervalo

false ∗ Se a data não estiver dentro do intervalo

5.1.2.22 isValidDataRange()

```
bool isValidDataRange (
            Data dataInicio,
            Data dataFim )
```

Função para verificar se as datas fornecidas formam um range válido.

Parâmetros

dataInicio	∗ Data de início
dataFim	∗ Data de fim

Retorna

true ∗ Se o range for válido

false ∗ Se o range não for válido

5.1.2.23 lerInteiro()

```
void lerInteiro (
            int * n )
```

Função para ler um número inteiro.

Parâmetros

n	* Número inteiro

5.1.2.24 liberarListaClientes()

```
void liberarListaClientes (
            ListaCliente * li )
```

Função para liberar a lista de clientes.

Parâmetros

li	* Lista de clientes

5.1.2.25 liberarListaDietasRealizadas()

```
void liberarListaDietasRealizadas (
            ListaDietaRealizada * li )
```

Função para liberar a lista de dietas realizadas.

Parâmetros

li	* Lista de dietas realizadas

5.1.2.26 liberarListaPlanosNutricionais()

```
void liberarListaPlanosNutricionais (
            ListaPlanoNutricional * li )
```

Função para liberar a lista de planos nutricionais.

Parâmetros

li	∗ Lista de planos nutricionais

5.1.2.27 liberarListaRefeicoesPlaneadas()

```
void liberarListaRefeicoesPlaneadas (
            ListaRefeicoesPlaneadas * li )
```

Função para liberar a lista de refeições planeadas.

Parâmetros

li	∗ Lista de refeições planeadas

5.1.2.28 ordenarClientesDecrescente()

```
void ordenarClientesDecrescente (
            ListaCliente * li )
```

Função para ordenar os clientes de forma decrescente.

Parâmetros

li	∗ Lista de clientes para ordenar

5.1.2.29 perguntarData()

```
Data perguntarData ( )
```

Função para perguntar a data ao utilizador.

Retorna

Data ∗ Data inserida pelo utilizador

5.1.2.30 verificarParametros()

```
bool verificarParametros (
            int argc,
            char * argv[] )
```

Função para verificar os parâmetros passados pelo utilizador.

Parâmetros

argc	∗ Nº de parametros passado pelo utilizador
argv	∗ Array de parametros passado pelo utilizador

Retorna

true ∗ Se o utilizador passou o parâmetro "-ajuda"

false ∗ Se o utilizador não passou o parâmetro "-ajuda"

5.2 Referência ao ficheiro /root/d-35/src/functions.h

```
#include <stdbool.h>
#include "structs.h"
```

Funções

- bool verificarParametros (int argc, char ∗argv[ ])

  Função para verificar os parâmetros passados pelo utilizador.
- void lerInteiro (int ∗n)

  Função para ler um número inteiro.
- ListaCliente ∗ criarListasClientes ()

  Função para criar a lista de clientes.
- ListaDietaRealizada ∗ criarListasDietasRealizadas ()

  Função para criar a lista de dietas realizadas.
- ListaPlanoNutricional ∗ criarListasPlanosNutricionais ()

  Função para criar a lista de planos nutricionais.
- ListaRefeicoesPlaneadas ∗ criarListasRefeicoesPlaneadas ()

  Função para criar a lista de refeições planeadas.
- bool criarListas ()

  Função para criar as listas na inicialização do programa.
- void liberarListaClientes (ListaCliente ∗li)

  Função para liberar a lista de clientes.
- void liberarListaDietasRealizadas (ListaDietaRealizada ∗li)

  Função para liberar a lista de dietas realizadas.
- void liberarListaPlanosNutricionais (ListaPlanoNutricional ∗li)

  Função para liberar a lista de planos nutricionais.
- void liberarListaRefeicoesPlaneadas (ListaRefeicoesPlaneadas ∗li)

  Função para liberar a lista de refeições planeadas.
- void endProgram ()

  Função para terminar o programa e apagar as listas.
- bool checkClienteID (ListaCliente ∗li, int id)

  Função para verificar se o ID do cliente existe.
- bool checkPlanoNutricionalByClienteDataRef (ListaPlanoNutricional ∗li, int idCliente, Data data, int refeicao)

  Função para verificar se já existe um plano nutricional para o cliente dentro do intervalo de tempo.
- bool insertCliente (ListaCliente ∗li, Cliente cliente)

  Função para inserir um novo cliente na lista.

- bool insertDietaRealizada (ListaDietaRealizada ∗li, DietaRealizada dietaRealizada)

  Função para inserir uma dieta realizada.
- bool insertPlanoNutricional (ListaPlanoNutricional ∗li, PlanoNutricional planoNutricional)

  Função para inserir um plano nutricional.
- bool insertRefeicaoPlaneada (ListaRefeicoesPlaneadas ∗li, RefeicoesPlaneadas refeicoesPlaneadas)

  Função para inserir uma refeição planeada.
- int convertRefeicao (char ∗refeicao)

  Função para converter a string de uma refeição para número.
- char ∗ convertRefeicaoString (int refeicao)

  Função para converter o número da refeição para a sua string.
- bool fileExists (char ∗fileName)

  Função para verificar se o arquivo existe.
- bool carregarClientes ()

  Função para carregar os clientes.
- bool carregarDietasRealizadas ()

  Função para carregar as dietas realizadas.
- bool carregarPlanosNutricionais ()

  Função para carregar os planos nutricionais.
- void displayClientes ()

  Função para fazer display dos clientes inseridos na lista.
- void displayDietasRealizadas ()

  Função para fazer display das dietas realizadas inseridas na lista.
- void displayPlanosNutricionais ()

  Função para fazer display dos planos nutricionais inseridos na lista.
- bool isAnoBissexto (int ano)

  Função para verificar se um ano é bissexto.
- Data perguntarData ()

  Função para perguntar a data ao utilizador.
- bool isValidDataRange (Data dataInicio, Data dataFim)

  Função para verificar se as datas fornecidas formam um range válido.
- bool isInDataRange (Data data, Data dataInicio, Data dataFim)

  Função para verificar se a data está dentro do intervalo de datas.
- bool iSDateRangeInsideOther (Data dataInicio, Data dataFim, Data dataInicio2, Data dataFim2)

  Função para verificar se um range de datas está dentro do outro.
- int diaMes (int mes, int ano)

  Função para obter os dias do mês.
- void ordenarClientesDecrescente (ListaCliente ∗li)

  Função para ordenar os clientes de forma decrescente.
- void clientesPassaramCaloriasPeriodoTempo ()

  Função para o ponto 2 do trabalho.
- void clientesPassaramPlanoNutricionalPeriodoTempoDecrescente ()

  Função para o ponto 3 do trabalho.
- void listarPlanoNutricionalPacienteRefeicaoPeriodoTempo ()

  Função para o ponto 4 do trabalho.
- void calcularMediaCalRefeicaoPassadoPeriodoTempo ()

  Função para o ponto 5 do trabalho.
- void gerarTabelaRefeicoesPlaneadas ()

  Função para o ponto 6 do trabalho.

5.2.1 Descrição detalhada

Versão

0.1

Data

2023-12-21

Copyright

Copyright (c) 2023

5.2.2 Documentação das funções

5.2.2.1 carregarClientes()

```
bool carregarClientes ( )
```

Função para carregar os clientes.

Retorna

true ∗ Se os clientes foram carregados com sucesso

false ∗ Se os clientes não foram carregados com sucesso

5.2.2.2 carregarDietasRealizadas()

```
bool carregarDietasRealizadas ( )
```

Função para carregar as dietas realizadas.

Retorna

true ∗ Se as dietas realizadas foram carregadas com sucesso

false ∗ Se as dietas realizadas não foram carregadas com sucesso

5.2.2.3 carregarPlanosNutricionais()

```
bool carregarPlanosNutricionais ( )
```

Função para carregar os planos nutricionais.

Retorna

true ∗ Se os planos nutricionais foram carregados com sucesso

false ∗ Se os planos nutricionais não foram carregados com sucesso

5.2.2.4 checkClienteID()

```
bool checkClienteID (
            ListaCliente * li,
            int id )
```

Função para verificar se o ID do cliente existe.

Parâmetros

li	∗ Lista de clientes
id	∗ ID do cliente

Retorna

true ∗ Se o ID do cliente existe

false ∗ Se o ID do cliente não existe

5.2.2.5 checkPlanoNutricionalByClienteDataRef()

```
bool checkPlanoNutricionalByClienteDataRef (
            ListaPlanoNutricional * li,
            int idCliente,
            Data data,
            int refeicao )
```

Função para verificar se já existe um plano nutricional para o cliente dentro do intervalo de tempo.

Parâmetros

li	∗ Lista de planos nutricionais
idCliente	∗ ID do cliente
data	∗ Data
refeicao	∗ Refeição

Retorna

true ∗ Se já existe um plano nutricional para o cliente dentro do intervalo de tempo

false ∗ Se não existe um plano nutricional para o cliente dentro do intervalo de tempo

5.2.2.6 convertRefeicao()

```
int convertRefeicao (
            char * refeicao )
```

Função para converter a string de uma refeição para número.

Parâmetros

refeicao	∗ String da refeição

Retorna

int ∗ Número da refeição

5.2.2.7 convertRefeicaoString()

```
char* convertRefeicaoString (
            int refeicao )
```

Função para converter o número da refeição para a sua string.

Parâmetros

refeicao	∗ Número da refeição

Retorna

char∗ ∗ String da refeição

5.2.2.8 criarListas()

```
bool criarListas ( )
```

Função para criar as listas na inicialização do programa.

Retorna

true ∗ Se as listas foram criadas

false ∗ Se as listas não foram criadas

5.2.2.9 criarListasClientes()

```
ListaCliente* criarListasClientes ( )
```

Função para criar a lista de clientes.

Retorna

ListaCliente** * Lista de clientes

5.2.2.10 criarListasDietasRealizadas()

```
ListaDietaRealizada* criarListasDietasRealizadas ( )
```

Função para criar a lista de dietas realizadas.

Retorna

ListaDietaRealizada** * Lista de dietas realizadas

5.2.2.11 criarListasPlanosNutricionais()

```
ListaPlanoNutricional* criarListasPlanosNutricionais ( )
```

Função para criar a lista de planos nutricionais.

Retorna

ListaPlanoNutricional** * Lista de planos nutricionais

5.2.2.12 criarListasRefeicoesPlaneadas()

```
ListaRefeicoesPlaneadas* criarListasRefeicoesPlaneadas ( )
```

Função para criar a lista de refeições planeadas.

Retorna

ListaRefeicoesPlaneadas** * Lista de refeições planeadas

5.2.2.13 diaMes()

```
int diaMes (
            int mes,
            int ano )
```

Função para obter os dias do mês.

Parâmetros

mes	∗ Mês
ano	∗ Ano

Retorna

int ∗ Número de dias do mês

5.2.2.14 fileExists()

```
bool fileExists (
            char * fileName )
```

Função para verificar se o arquivo existe.

Parâmetros

fileName	∗ Nome/Caminho do arquivo

Retorna

true ∗ Se o arquivo existir

false ∗ Se o arquivo não existir

5.2.2.15 insertCliente()

```
bool insertCliente (
            ListaCliente * li,
            Cliente cliente )
```

Função para inserir um novo cliente na lista.

Parâmetros

li	∗ Lista de clientes
cliente	∗ Dados do cliente

Retorna

true ∗ Se o cliente foi inserido com sucesso

false ∗ Se o cliente não foi inserido com sucesso

5.2.2.16 insertDietaRealizada()

```
bool insertDietaRealizada (
            ListaDietaRealizada * li,
            DietaRealizada dietaRealizada )
```

Função para inserir uma dieta realizada.

Parâmetros

li	∗ Lista de dietas realizadas
dietaRealizada	∗ Dados da dieta realizada

Retorna

true ∗ Se a dieta realizada foi inserida com sucesso

false ∗ Se a dieta realizada não foi inserida com sucesso

5.2.2.17 insertPlanoNutricional()

```
bool insertPlanoNutricional (
            ListaPlanoNutricional * li,
            PlanoNutricional planoNutricional )
```

Função para inserir um plano nutricional.

Parâmetros

li	∗ Lista de planos nutricionais
planoNutricional	∗ Dados do plano nutricional

Retorna

true ∗ Se o plano nutricional foi inserido com sucesso

false ∗ Se o plano nutricional não foi inserido com sucesso

5.2.2.18 insertRefeicaoPlaneada()

```
bool insertRefeicaoPlaneada (
            ListaRefeicoesPlaneadas * li,
            RefeicoesPlaneadas refeicoesPlaneadas )
```

Função para inserir uma refeição planeada.

Parâmetros

li	∗ Lista de refeições planeadas
refeicoesPlaneadas	∗ Dados da refeição planeada

Retorna

true ∗ Se a refeição planeada foi inserida com sucesso

false ∗ Se a refeição planeada não foi inserida com sucesso

5.2.2.19 isAnoBissexto()

```
bool isAnoBissexto (
            int ano )
```

Função para verificar se um ano é bissexto.

Parâmetros

ano	∗ Ano

Retorna

true ∗ Se o ano for bissexto

false ∗ Se o ano não for bissexto

5.2.2.20 iSDateRangeInsideOther()

```
bool iSDateRangeInsideOther (
            Data dataInicio,
            Data dataFim,
            Data dataInicio2,
            Data dataFim2 )
```

Função para verificar se um range de datas está dentro do outro.

Parâmetros

dataInicio	∗ Data de início do 1º range
dataFim	∗ Data de fim do 1º range
dataInicio2	∗ Data de início do 2º range
dataFim2	∗ Data de fim do 2º range

Retorna

true ∗ Se o 1º range estiver dentro do 2º range

false ∗ Se o 1º range não estiver dentro do 2º range

5.2.2.21 isInDataRange()

```
bool isInDataRange (
            Data data,
            Data dataInicio,
            Data dataFim )
```

Função para verificar se a data está dentro do intervalo de datas.

Parâmetros

data	∗ Data para verificar
dataInicio	∗ Data de início
dataFim	∗ Data de fim

Retorna

true ∗ Se a data estiver dentro do intervalo

false ∗ Se a data não estiver dentro do intervalo

5.2.2.22 isValidDataRange()

```
bool isValidDataRange (
            Data dataInicio,
            Data dataFim )
```

Função para verificar se as datas fornecidas formam um range válido.

Parâmetros

dataInicio	∗ Data de início
dataFim	∗ Data de fim

Retorna

true ∗ Se o range for válido

false ∗ Se o range não for válido

5.2.2.23 lerInteiro()

```
void lerInteiro (
            int * n )
```

Função para ler um número inteiro.

Parâmetros

n	* Número inteiro

5.2.2.24 liberarListaClientes()

```
void liberarListaClientes (
            ListaCliente * li )
```

Função para liberar a lista de clientes.

Parâmetros

li	* Lista de clientes

5.2.2.25 liberarListaDietasRealizadas()

```
void liberarListaDietasRealizadas (
            ListaDietaRealizada * li )
```

Função para liberar a lista de dietas realizadas.

Parâmetros

li	* Lista de dietas realizadas

5.2.2.26 liberarListaPlanosNutricionais()

```
void liberarListaPlanosNutricionais (
            ListaPlanoNutricional * li )
```

Função para liberar a lista de planos nutricionais.

Parâmetros

li	∗ Lista de planos nutricionais

5.2.2.27 liberarListaRefeicoesPlaneadas()

```
void liberarListaRefeicoesPlaneadas (
            ListaRefeicoesPlaneadas * li )
```

Função para liberar a lista de refeições planeadas.

Parâmetros

li	∗ Lista de refeições planeadas

5.2.2.28 ordenarClientesDecrescente()

```
void ordenarClientesDecrescente (
            ListaCliente * li )
```

Função para ordenar os clientes de forma decrescente.

Parâmetros

li	∗ Lista de clientes para ordenar

5.2.2.29 perguntarData()

```
Data perguntarData ( )
```

Função para perguntar a data ao utilizador.

Retorna

Data ∗ Data inserida pelo utilizador

5.2.2.30 verificarParametros()

```
bool verificarParametros (
            int argc,
            char * argv[] )
```

Função para verificar os parâmetros passados pelo utilizador.

Parâmetros

argc	∗ Nº de parametros passado pelo utilizador
argv	∗ Array de parametros passado pelo utilizador

Retorna

true ∗ Se o utilizador passou o parâmetro "-ajuda"

false ∗ Se o utilizador não passou o parâmetro "-ajuda"

5.3 Referência ao ficheiro /root/d-35/src/main.c

```
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>
#include "menus.h"
#include "functions.h"
```

Funções

- int **main** (int argc, char ∗argv[ ])

5.3.1 Descrição detalhada

Versão

0.1

Data

2023-12-21

Copyright

Copyright (c) 2023

5.4 Referência ao ficheiro /root/d-35/src/menus.c

```
#include <stdio.h>
#include <stdlib.h>
#include "menus.h"
#include "functions.h"
```

Funções

- void menuPrincipal ()

  Função para mostrar o menu principal.
- void menuCarregarDados ()

  Função para mostrar o menu para carregar dados.
- void menuAjuda ()

  Função para mostrar o menu de ajuda.
- void menuDisplayDados ()

  Função para mostrar o menu de visualização de dados.

5.4.1 Descrição detalhada

Versão

0.1

Data

2023-12-21

Copyright

Copyright (c) 2023

5.4.2 Documentação das funções

5.4.2.1 menuAjuda()

```
void menuAjuda ( )
```

Função para mostrar o menu de ajuda.

Retorna

void

5.4.2.2 menuCarregarDados()

```
void menuCarregarDados ( )
```

Função para mostrar o menu para carregar dados.

Retorna

void

5.4.2.3 menuDisplayDados()

```
void menuDisplayDados ( )
```

Função para mostrar o menu de visualização de dados.

Retorna

void

5.4.2.4 menuPrincipal()

```
void menuPrincipal ( )
```

Função para mostrar o menu principal.

Retorna

void

5.5 Referência ao ficheiro /root/d-35/src/menus.h

Funções

- void menuPrincipal ()
  Função para mostrar o menu principal.
- void menuCarregarDados ()
  Função para mostrar o menu para carregar dados.
- void menuAjuda ()
  Função para mostrar o menu de ajuda.
- void menuDisplayDados ()
  Função para mostrar o menu de visualização de dados.

5.5.1 Descrição detalhada

Versão

0.1

Data

2023-12-21

Copyright

Copyright (c) 2023

5.5.2 Documentação das funções

5.5.2.1 menuAjuda()

```
void menuAjuda ( )
```

Função para mostrar o menu de ajuda.

Retorna

void

5.5.2.2 menuCarregarDados()

```
void menuCarregarDados ( )
```

Função para mostrar o menu para carregar dados.

Retorna

void

5.5.2.3 menuDisplayDados()

```
void menuDisplayDados ( )
```

Função para mostrar o menu de visualização de dados.

Retorna

void

5.5.2.4 menuPrincipal()

```
void menuPrincipal ( )
```

Função para mostrar o menu principal.

Retorna

void

5.6 Referência ao ficheiro /root/d-35/src/structs.h

Componentes

- struct Data
- struct Cliente
- struct DietaRealizada
- struct PlanoNutricional
- struct RefeicoesPlaneadas
- struct elementoCliente
- struct elementoDietaRelizada
- struct elementoPlanoNutricional
- struct elementoRefeicoesPlaneadas

Definições de tipos

- typedef struct elementoCliente **ElementoCliente**
- typedef struct elementoDietaRelizada **ElementoDietaRealizada**
- typedef struct elementoPlanoNutricional **ElementoPlanoNutricional**
- typedef struct elementoRefeicoesPlaneadas **ElementoRefeicoesPlaneadas**
- typedef ElementoCliente * **ListaCliente**
- typedef ElementoDietaRealizada * **ListaDietaRealizada**
- typedef ElementoPlanoNutricional * **ListaPlanoNutricional**
- typedef ElementoRefeicoesPlaneadas * **ListaRefeicoesPlaneadas**

5.6.1 Descrição detalhada

Versão

0.1

Data

2023-12-21

Copyright

Copyright (c) 2023