Trabalho-PI 2023/2024

Gerado por Doxygen 1.9.5

Capítulo 1

Índice das estruturas de dados

1.1 Estruturas de dados

Lista das estruturas de dados com uma breve descrição:

Capítulo 2

Índice dos ficheiros

2.1 Lista de ficheiros

Lista de todos os ficheiros com uma breve descrição:

Capítulo 3

Documentação da estruturas de dados

3.1 Referência à estrutura Date

Estrutura para representar uma data.

`#include <types.h>`

Campos de Dados

- int day
- int month
- int year

3.1.1 Descrição detalhada

Estrutura para representar uma data.

Esta estrutura é utilizada para representar uma data, incluindo dia, mês e ano. É ideal para armazenar e manipular datas em diversos contextos dentro do programa, como datas de nascimento, eventos ou outros marcos temporais relevantes.

3.1.2 Documentação dos campos e atributos

3.1.2.1 day

`Date::day`

Membro 'day' representa o dia do mês. Deve ser um valor entre 1 e 31, dependendo do mês e do ano.

Referenciado por dateInPeriod(), fillPeriod(), listMealPlan(), printDate() e printTable().

3.1.2.2 month

`Date::month`

Membro 'month' representa o mês do ano. Deve ser um valor entre 1 (Janeiro) e 12 (Dezembro).

Referenciado por dateInPeriod(), fillPeriod(), listMealPlan(), printDate(), printTable() e readFile().

3.1.2.3 year

`Date::year`

Membro 'year' representa o ano. Deve ser um valor inteiro que representa o ano, como 2023.

Referenciado por dateInPeriod(), fillPeriod(), listMealPlan(), printDate(), printTable() e readFile().

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

- src/types.h

3.2 Referência à estrutura Diet

Estrutura para representar informações sobre a dieta de um paciente.

`#include <types.h>`

Campos de Dados

- int ID
- Date date
- char meal [50]
- char food [50]
- int calories

3.2.1 Descrição detalhada

Estrutura para representar informações sobre a dieta de um paciente.

Esta estrutura é utilizada para armazenar detalhes sobre as refeições consumidas por um paciente, incluindo a data da refeição, o tipo de refeição, os alimentos consumidos e o total de calorias. É essencial para sistemas de gestão de saúde ou nutrição, onde é necessário monitorizar e analisar a dieta e o consumo calórico dos pacientes.

3.2.2 Documentação dos campos e atributos

3.2.2.1 calories

`Diet::calories`

Membro 'calories' representa o total de calorias consumidas na refeição. É um valor inteiro.

Referenciado por averageCalories(), exceededCalories(), printTable() e readFile().

3.2.2.2 date

`Diet::date`

Membro 'date' armazena a data em que a refeição foi consumida. É uma estrutura 'Date' que inclui dia, mês e ano.

Referenciado por readFile().

3.2.2.3 food

`Diet::food`

Membro 'food' descreve os alimentos consumidos na refeição. É uma cadeia de caracteres com um tamanho máximo de 50 caracteres.

Referenciado por readFile().

3.2.2.4 ID

`Diet::ID`

Membro 'ID' representa um identificador único para o paciente associado a esta entrada da dieta. É um valor inteiro.

Referenciado por exceededCalories(), initializeDiets(), outOfRange() e printTable().

3.2.2.5 meal

`Diet::meal`

Membro 'meal' armazena o tipo de refeição (por exemplo, 'almoço', 'jantar'). É uma cadeia de caracteres com um tamanho máximo de 50 caracteres.

Referenciado por readFile().

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

- src/types.h

3.3 Referência à estrutura IDCalories

Estrutura para representar o consumo de calorias associado a um identificador de paciente.

`#include <types.h>`

Campos de Dados

- int ID
- int calories

3.3.1 Descrição detalhada

Estrutura para representar o consumo de calorias associado a um identificador de paciente.

Esta estrutura é usada para associar um identificador de paciente (ID) com a quantidade de calorias consumidas. É particularmente útil em sistemas de monitorização nutricional ou de saúde, onde é necessário acompanhar e analisar o consumo calórico dos pacientes. A estrutura permite uma rápida correlação entre o paciente e suas calorias consumidas, facilitando o processamento e análise desses dados.

3.3.2 Documentação dos campos e atributos

3.3.2.1 calories

`IDCalories::calories`

Membro 'calories' armazena o total de calorias consumidas pelo paciente. É um valor inteiro que representa o consumo calórico.

Referenciado por exceededCalories().

3.3.2.2 ID

`IDCalories::ID`

Membro 'ID' representa um identificador único do paciente. É um valor inteiro.

Referenciado por exceededCalories().

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

- src/types.h

3.4 Referência à estrutura InfoTable

Estrutura para representar informações detalhadas de um plano alimentar e o consumo real de um paciente.

`#include <types.h>`

Campos de Dados

- Patients patient
- char meal [50]
- Period period
- int minCal
- int maxCal
- int calories

3.4.1 Descrição detalhada

Estrutura para representar informações detalhadas de um plano alimentar e o consumo real de um paciente.

Esta estrutura é usada para combinar informações sobre um paciente com detalhes específicos de uma refeição, incluindo o período em que a refeição está planeada, o intervalo calórico recomendado (mínimo e máximo) e o total de calorias efetivamente consumidas. Serve como um registro abrangente para comparar o plano alimentar proposto com o consumo real, permitindo análises e ajustes no acompanhamento nutricional do paciente.

3.4.2 Documentação dos campos e atributos

3.4.2.1 calories

`InfoTable::calories`

Membro 'calories' representa o total de calorias consumidas pelo paciente na refeição. É um valor inteiro que ajuda a monitorizar a adesão ao plano alimentar.

Referenciado por printTable().

3.4.2.2 maxCal

`InfoTable::maxCal`

Membro 'maxCal' especifica o limite máximo de calorias recomendado para a refeição. É um valor inteiro.

Referenciado por printTable().

3.4.2.3 meal

`InfoTable::meal`

Membro 'meal' representa o tipo de refeição planeada (por exemplo, 'almoço', 'jantar'). É uma cadeia de caracteres com um tamanho máximo de 50 caracteres.

Referenciado por printTable().

3.4.2.4 minCal

`InfoTable::minCal`

Membro 'minCal' especifica o limite mínimo de calorias recomendado para a refeição. É um valor inteiro.

Referenciado por printTable().

3.4.2.5 patient

`InfoTable::patient`

Membro 'patient' armazena informações sobre o paciente. É uma estrutura 'Patients' que inclui identificador, nome e número de telefone.

Referenciado por printTable().

3.4.2.6 period

`InfoTable::period`

Membro 'period' define o período de tempo para o qual o plano alimentar é válido. É uma estrutura 'Period' que inclui datas de início e fim.

Referenciado por printTable().

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

- src/types.h

3.5 Referência à estrutura MealPlan

Estrutura para representar um plano alimentar para um paciente.

`#include <types.h>`

Campos de Dados

- int ID
- Date date
- char meal [50]
- int minCal
- int maxCal

3.5.1 Descrição detalhada

Estrutura para representar um plano alimentar para um paciente.

Esta estrutura é utilizada para armazenar informações detalhadas sobre um plano alimentar específico de um paciente. Inclui um identificador do paciente, a data da refeição planeada, o tipo de refeição, e os limites de calorias (mínimo e máximo) estipulados para essa refeição. É essencial para a gestão e monitorização de planos nutricionais em contextos clínicos ou de saúde e bem-estar, permitindo o acompanhamento e a adequação da ingestão calórica às necessidades individuais de cada paciente.

3.5.2 Documentação dos campos e atributos

3.5.2.1 date

`MealPlan::date`

Membro 'date' armazena a data em que a refeição planeada deve ser consumida. É uma estrutura 'Date' que inclui dia, mês e ano.

Referenciado por listMealPlan(), printTable() e readFile().

3.5.2.2 ID

`MealPlan::ID`

Membro 'ID' representa o identificador único do paciente a quem o plano alimentar se destina. É um valor inteiro.

Referenciado por initializeMealPlans() e printTable().

3.5.2.3 maxCal

`MealPlan::maxCal`

Membro 'maxCal' define o limite máximo de calorias recomendado para a refeição. É um valor inteiro.

Referenciado por listMealPlan(), printTable() e readFile().

3.5.2.4 meal

`MealPlan::meal`

Membro 'meal' armazena o tipo de refeição planeada (por exemplo, 'almoço', 'jantar'). É uma cadeia de caracteres com um tamanho máximo de 50 caracteres.

Referenciado por readFile().

3.5.2.5 minCal

`MealPlan::minCal`

Membro 'minCal' define o limite mínimo de calorias recomendado para a refeição. É um valor inteiro.

Referenciado por listMealPlan(), printTable() e readFile().

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

- src/types.h

3.6 Referência à estrutura Patients

Estrutura para representar as informações de um paciente.

`#include <types.h>`

Campos de Dados

- int ID
- char name [50]
- int phoneNumber

3.6.1 Descrição detalhada

Estrutura para representar as informações de um paciente.

Esta estrutura é utilizada para armazenar e manipular dados relacionados aos pacientes. Inclui identificador único, nome e número de telefone. É essencial para gerir informações de pacientes em contextos como clínicas, hospitais, ou qualquer sistema que necessite manter um registo detalhado dos pacientes.

3.6.2 Documentação dos campos e atributos

3.6.2.1 ID

`Patients::ID`

Membro 'ID' representa um identificador único para o paciente. É um valor inteiro.

Referenciado por initializePatients() e printTable().

3.6.2.2 name

`Patients::name`

Membro 'name' armazena o nome do paciente. É uma cadeia de caracteres com um tamanho máximo de 50 caracteres.

Referenciado por printTable().

3.6.2.3 phoneNumber

`Patients::phoneNumber`

Membro 'phoneNumber' armazena o número de telefone do paciente. É um valor inteiro que representa o número de telefone.

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

- src/types.h

3.7 Referência à estrutura Period

Estrutura para representar um período de tempo definido por uma data de início e uma data de fim.

```
#include <types.h>
```

Campos de Dados

- Date begin
- Date end

3.7.1 Descrição detalhada

Estrutura para representar um período de tempo definido por uma data de início e uma data de fim.

Esta estrutura é usada para definir um intervalo de tempo, especificando uma data de início e uma data de fim. Cada uma destas datas é representada por uma estrutura 'Date', que inclui dia, mês e ano. Este tipo de estrutura é particularmente útil para representar períodos de tempo em aplicações que necessitam de gerir eventos, intervalos de disponibilidade, períodos de validade, entre outros.

3.7.2 Documentação dos campos e atributos

3.7.2.1 begin

`Period::begin`

Membro 'begin' é uma estrutura 'Date' que representa a data de início do período.

Referenciado por dateInPeriod(), fillPeriod(), listMealPlan(), printPeriod() e printTable().

3.7.2.2 end

`Period::end`

Membro 'end' é uma estrutura 'Date' que representa a data de fim do período.

Nota

É importante garantir a coerência das datas, onde a data de início deve ser anterior ou igual à data de fim.

Referenciado por dateInPeriod(), fillPeriod(), listMealPlan(), printPeriod() e printTable().

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

- src/types.h

Capítulo 4

Documentação do ficheiro

4.1 Referência ao ficheiro src/logic.c

Implementação das funções de lógica principal do programa.

```
#include "logic.h"
#include "utils.h"
#include <stdio.h>
#include <string.h>
```

Funções

- int exceededCalories (Diet *diet, int max_size, int calories, Period period)

  Calcula o número de pacientes que excederam um limite de calorias num determinado período.
- int outOfRange (Diet *diet, MealPlan *mealPlan, Period period, int max_size)

  Calcula o número de pacientes cuja ingestão calórica está fora do intervalo definido no seu plano de refeições.
- int listMealPlan (MealPlan *mealPlan, Period period, int max_size, char *mealType, int IDNum)

  Lista as refeições de um plano alimentar para um paciente específico num dado período.
- float averageCalories (Diet *diet, Period period, int max_size, char *mealType, int IDNum)

  Calcula a média de calorias consumidas por um paciente num tipo específico de refeição durante um período.
- void printTable (MealPlan *mealPlans, Diet *diets, Patients *patients, int max_size)

  Imprime uma tabela com o resumo dos planos alimentares e o consumo calórico dos pacientes.

4.1.1 Descrição detalhada

Implementação das funções de lógica principal do programa.

Este ficheiro contém as definições das funções principais que lidam com a lógica do programa. Inclui funções para calcular calorias excedidas, verificar refeições fora do intervalo de calorias estipulado, listar planos de refeições, calcular a média de calorias consumidas e imprimir tabelas com informações relevantes.

As funções neste ficheiro operam sobre as estruturas de dados definidas em 'logic.h' e 'utils.h', sendo essenciais para as operações principais do programa, como a gestão de dietas e planos alimentares.

Nota

Este ficheiro faz uso extensivo das estruturas e funções definidas em 'utils.h' para operações de data e manipulação de dados.

Aviso

Algumas funções assumem que os dados de entrada são válidos e não realizam verificações extensivas de erro. A validação adequada dos dados de entrada é responsabilidade das funções chamadoras.

4.1.2 Documentação das funções

4.1.2.1 averageCalories()

```
float averageCalories (
            Diet * diet,
            Period period,
            int max_size,
            char * mealType,
            int IDNum )
```

Calcula a média de calorias consumidas por um paciente num tipo específico de refeição durante um período.

Esta função percorre um array de estruturas 'Diet', somando e contabilizando as calorias consumidas pelo paciente especificado, para um tipo específico de refeição, dentro do período definido. A média de calorias é calculada com base no total de calorias consumidas e no número de refeições contabilizadas.

Parâmetros

diet	Ponteiro para o array de estruturas 'Diet', que contém os dados de consumo de calorias dos pacientes.
period	Estrutura 'Period' que define o período de tempo durante o qual o consumo é avaliado.
max_size	Tamanho máximo do array 'diet'.
mealType	String que especifica o tipo de refeição a ser considerada no cálculo (ex: "almoço", "jantar").
IDNum	Identificador numérico do paciente cuja média de calorias será calculada.

Retorna

Retorna a média de calorias consumidas pelo paciente para o tipo de refeição especificado no período dado. Se não forem encontradas refeições correspondentes, retorna 0.0.

Nota

Esta função pressupõe que o array 'diet' é válido, e que o tamanho máximo do array e a estrutura 'period' são respeitados. Além disso, assume-se que a string 'mealType' é válida.

Aviso

A função não verifica se a string 'mealType' corresponde a um tipo de refeição existente no array 'diet', dependendo da consistência dos dados fornecidos.

Referências Diet::calories e dateInPeriod().

Referenciado por handleAverageCalories().

4.1.2.2 exceededCalories()

```
int exceededCalories (
            Diet * diet,
            int max_size,
            int calories,
            Period period )
```

Calcula o número de pacientes que excederam um limite de calorias num determinado período.

Esta função percorre um array de estruturas 'Diet' e identifica quantos pacientes consumiram mais calorias do que o limite especificado durante um período de tempo definido. Usa um array auxiliar 'idCalories' para armazenar e somar as calorias consumidas por cada paciente.

Parâmetros

diet	Ponteiro para o array de estruturas 'Diet', que contém os dados de consumo de calorias.
max_size	Tamanho máximo do array 'diet'.
calories	Limite de calorias a ser considerado para determinar o excesso de consumo.
period	Estrutura 'Period' que define o período de tempo durante o qual o consumo é avaliado.

Retorna

Retorna o número de pacientes que excederam o limite de calorias no período especificado. Retorna -1 se o número de pacientes únicos exceder 100, indicando que o limite do array foi atingido.

Nota

Esta função pressupõe que o array 'diet' e a estrutura 'period' são válidos e que o tamanho máximo do array 'diet' é respeitado nas chamadas da função.

Aviso

A função utiliza um array fixo de tamanho 100 para rastrear os pacientes. Se houver mais de 100 pacientes únicos no período especificado, a função não poderá rastreá-los todos e retornará -1.

Referências Diet::calories, IDCalories::calories, dateInPeriod(), Diet::ID e IDCalories::ID.

Referenciado por handleExceededCalories().

4.1.2.3 listMealPlan()

```
int listMealPlan (
            MealPlan * mealPlan,
            Period period,
            int max_size,
            char * mealType,
            int IDNum )
```

Lista as refeições de um plano alimentar para um paciente específico num dado período.

Esta função percorre um array de estruturas 'MealPlan', listando as refeições que correspondem a um determinado tipo e que foram planeadas para um paciente específico durante um período definido. A função imprime os detalhes de cada refeição encontrada, incluindo datas e intervalos calóricos.

Parâmetros

mealPlan	Ponteiro para o array de estruturas 'MealPlan', representando o plano de refeições.
period	Estrutura 'Period' que define o período de tempo durante o qual as refeições são listadas.
max_size	Tamanho máximo do array 'mealPlan'.
mealType	String que representa o tipo de refeição a ser listada (ex: "almoço", "jantar").
IDNum	Identificador numérico do paciente para o qual as refeições serão listadas.

Retorna

Retorna o número de refeições listadas que correspondem aos critérios especificados.

Nota

Esta função pressupõe que o array 'mealPlan' é válido e que o tamanho máximo do array é respeitado. Além disso, assume-se que a string 'mealType' e a estrutura 'period' são válidas.

Aviso

A função imprime diretamente para o standard output e não realiza a formatação avançada ou a paginação dos resultados, podendo ser menos adequada para grandes conjuntos de dados ou para a integração em interfaces de utilizador mais complexas.

Referências Period::begin, MealPlan::date, dateInPeriod(), Date::day, Period::end, MealPlan::maxCal, MealPlan::minCal, Date::month e Date::year.

Referenciado por handleMealPlan().

4.1.2.4 outOfRange()

```
int outOfRange (
            Diet * diet,
            MealPlan * mealPlan,
            Period period,
            int max_size )
```

Calcula o número de pacientes cuja ingestão calórica está fora do intervalo definido no seu plano de refeições.

Esta função compara o consumo calórico dos pacientes, registrado no array 'diet', com os intervalos calóricos definidos no seu plano de refeições, 'mealPlan', durante um determinado período. Os IDs dos pacientes cuja ingestão calórica esteja fora do intervalo são armazenados e contados.

Parâmetros

diet	Ponteiro para o array de estruturas 'Diet', que contém os dados de consumo de calorias dos pacientes.
mealPlan	Ponteiro para o array de estruturas 'MealPlan', que define os intervalos calóricos para os pacientes.
period	Estrutura 'Period' que define o período de tempo durante o qual o consumo é avaliado.
max_size	Tamanho máximo dos arrays 'diet' e 'mealPlan'.

Retorna

Retorna o número de pacientes cujo consumo de calorias está fora do intervalo estipulado no seu plano de refeições.

Nota

Esta função pressupõe que os arrays 'diet' e 'mealPlan' são válidos, e que o tamanho máximo dos arrays é respeitado. Além disso, assume-se que os IDs dos pacientes são únicos.

Aviso

A função utiliza um array fixo de tamanho 100 para armazenar os IDs dos pacientes. Se houver mais de 100 pacientes únicos no período especificado, a função não poderá rastrear todos e os resultados podem não ser completos.

Referências dateInPeriod(), Diet::ID e sortDescending().

Referenciado por handleOutOfRange().

4.1.2.5 printTable()

```
void printTable (
            MealPlan * mealPlans,
            Diet * diets,
            Patients * patients,
            int max_size )
```

Imprime uma tabela com o resumo dos planos alimentares e o consumo calórico dos pacientes.

Esta função constrói e imprime uma tabela detalhada que mostra o plano alimentar de cada paciente, incluindo os tipos de refeição, o período de cada plano, as calorias mínimas e máximas estipuladas, e o total de calorias consumidas. A função itera sobre os arrays 'mealPlans' e 'diets', preenchendo uma estrutura auxiliar 'infoTable' para armazenar as informações consolidadas antes de imprimir.

Parâmetros

mealPlans	Ponteiro para o array de estruturas 'MealPlan', representando os planos alimentares.
diets	Ponteiro para o array de estruturas 'Diet', representando o consumo de calorias dos pacientes.
patients	Ponteiro para o array de estruturas 'Patients', contendo informações sobre os pacientes.
max_size	Tamanho máximo dos arrays 'mealPlans', 'diets' e 'patients'.

Nota

Esta função pressupõe que os arrays 'mealPlans', 'diets' e 'patients' são válidos e que o tamanho máximo dos arrays é respeitado. Além disso, assume-se que os IDs dos pacientes são únicos.

Aviso

A função utiliza um array fixo 'infoTable' de tamanho 100 para armazenar as informações consolidadas. Se houver mais de 100 pacientes únicos, a tabela não poderá representar todos eles.

Referências Period::begin, Diet::calories, InfoTable::calories, MealPlan::date, dateInPeriod(), Date::day, Period::end, Patients::ID, Diet::ID, MealPlan::ID, MealPlan::maxCal, InfoTable::maxCal, InfoTable::meal, MealPlan::minCal, InfoTable::minCal, Date::month, Patients::name, InfoTable::patient, InfoTable::period e Date::year.

Referenciado por handlePrintTable().

4.2 Referência ao ficheiro src/logic.h

Cabeçalho das funções principais de lógica do programa.

```
#include "types.h"
```

Funções

- void printTable (MealPlan *mealPlans, Diet *diets, Patients *patients, int max_size)

  Imprime uma tabela com o resumo dos planos alimentares e o consumo calórico dos pacientes.
- int exceededCalories (Diet *diet, int max_size, int calories, Period period)

  Calcula o número de pacientes que excederam um limite de calorias num determinado período.
- int outOfRange (Diet *diet, MealPlan *mealPlan, Period period, int max_size)

  Calcula o número de pacientes cuja ingestão calórica está fora do intervalo definido no seu plano de refeições.
- int listMealPlan (MealPlan *mealPlan, Period period, int max_size, char *mealType, int IDNum)

  Lista as refeições de um plano alimentar para um paciente específico num dado período.
- float averageCalories (Diet *diet, Period period, int max_size, char *mealType, int IDNum)

  Calcula a média de calorias consumidas por um paciente num tipo específico de refeição durante um período.

4.2.1 Descrição detalhada

Cabeçalho das funções principais de lógica do programa.

Este ficheiro de cabeçalho define as interfaces das funções principais relacionadas à lógica do programa. Inclui funções para gerir os planos alimentares dos pacientes, calcular a ingestão calórica, listar e comparar refeições planeadas, entre outras operações essenciais. As funções aqui declaradas são cruciais para a execução das principais funcionalidades do programa, lidando com a manipulação e análise de dados relacionados a dietas e planos alimentares.

As funções declaradas neste ficheiro incluem:

- Impressão de tabelas com resumos de planos alimentares e consumo calórico.
- Preenchimento de períodos com datas de início e fim.
- Cálculo do número de pacientes que excederam um limite de calorias.
- Verificação do consumo calórico em relação aos planos alimentares.
- Listagem de refeições conforme critérios específicos.
- Cálculo da média de calorias consumidas por um paciente.

Nota

Este ficheiro depende das definições das estruturas de dados em 'types.h'.

4.2.2 Documentação das funções

4.2.2.1 averageCalories()

```
float averageCalories (
            Diet * diet,
            Period period,
            int max_size,
            char * mealType,
            int IDNum )
```

Calcula a média de calorias consumidas por um paciente num tipo específico de refeição durante um período.

Esta função percorre um array de estruturas 'Diet', somando e contabilizando as calorias consumidas pelo paciente especificado, para um tipo específico de refeição, dentro do período definido. A média de calorias é calculada com base no total de calorias consumidas e no número de refeições contabilizadas.

Parâmetros

diet	Ponteiro para o array de estruturas 'Diet', que contém os dados de consumo de calorias dos pacientes.
period	Estrutura 'Period' que define o período de tempo durante o qual o consumo é avaliado.
max_size	Tamanho máximo do array 'diet'.
mealType	String que especifica o tipo de refeição a ser considerada no cálculo (ex: "almoço", "jantar").
IDNum	Identificador numérico do paciente cuja média de calorias será calculada.

Retorna

Retorna a média de calorias consumidas pelo paciente para o tipo de refeição especificado no período dado. Se não forem encontradas refeições correspondentes, retorna 0.0.

Nota

Esta função pressupõe que o array 'diet' é válido, e que o tamanho máximo do array e a estrutura 'period' são respeitados. Além disso, assume-se que a string 'mealType' é válida.

Aviso

A função não verifica se a string 'mealType' corresponde a um tipo de refeição existente no array 'diet', dependendo da consistência dos dados fornecidos.

Referências Diet::calories e dateInPeriod().

Referenciado por handleAverageCalories().

4.2.2.2 exceededCalories()

```
int exceededCalories (
            Diet * diet,
            int max_size,
            int calories,
            Period period )
```

Calcula o número de pacientes que excederam um limite de calorias num determinado período.

Esta função percorre um array de estruturas 'Diet' e identifica quantos pacientes consumiram mais calorias do que o limite especificado durante um período de tempo definido. Usa um array auxiliar 'idCalories' para armazenar e somar as calorias consumidas por cada paciente.

Parâmetros

diet	Ponteiro para o array de estruturas 'Diet', que contém os dados de consumo de calorias.
max_size	Tamanho máximo do array 'diet'.
calories	Limite de calorias a ser considerado para determinar o excesso de consumo.
period	Estrutura 'Period' que define o período de tempo durante o qual o consumo é avaliado.

Retorna

Retorna o número de pacientes que excederam o limite de calorias no período especificado. Retorna -1 se o número de pacientes únicos exceder 100, indicando que o limite do array foi atingido.

Nota

Esta função pressupõe que o array 'diet' e a estrutura 'period' são válidos e que o tamanho máximo do array 'diet' é respeitado nas chamadas da função.

Aviso

A função utiliza um array fixo de tamanho 100 para rastrear os pacientes. Se houver mais de 100 pacientes únicos no período especificado, a função não poderá rastreá-los todos e retornará -1.

Referências Diet::calories, IDCalories::calories, dateInPeriod(), Diet::ID e IDCalories::ID.

Referenciado por handleExceededCalories().

4.2.2.3 listMealPlan()

```
int listMealPlan (
            MealPlan * mealPlan,
            Period period,
            int max_size,
            char * mealType,
            int IDNum )
```

Lista as refeições de um plano alimentar para um paciente específico num dado período.

Esta função percorre um array de estruturas 'MealPlan', listando as refeições que correspondem a um determinado tipo e que foram planeadas para um paciente específico durante um período definido. A função imprime os detalhes de cada refeição encontrada, incluindo datas e intervalos calóricos.

Parâmetros

mealPlan	Ponteiro para o array de estruturas 'MealPlan', representando o plano de refeições.
period	Estrutura 'Period' que define o período de tempo durante o qual as refeições são listadas.
max_size	Tamanho máximo do array 'mealPlan'.
mealType	String que representa o tipo de refeição a ser listada (ex: "almoço", "jantar").
IDNum	Identificador numérico do paciente para o qual as refeições serão listadas.

Retorna

Retorna o número de refeições listadas que correspondem aos critérios especificados.

Nota

Esta função pressupõe que o array 'mealPlan' é válido e que o tamanho máximo do array é respeitado. Além disso, assume-se que a string 'mealType' e a estrutura 'period' são válidas.

Aviso

A função imprime diretamente para o standard output e não realiza a formatação avançada ou a paginação dos resultados, podendo ser menos adequada para grandes conjuntos de dados ou para a integração em interfaces de utilizador mais complexas.

Referências Period::begin, MealPlan::date, dateInPeriod(), Date::day, Period::end, MealPlan::maxCal, MealPlan::minCal, Date::month e Date::year.

Referenciado por handleMealPlan().

4.2.2.4 outOfRange()

```
int outOfRange (
            Diet * diet,
            MealPlan * mealPlan,
            Period period,
            int max_size )
```

Calcula o número de pacientes cuja ingestão calórica está fora do intervalo definido no seu plano de refeições.

Esta função compara o consumo calórico dos pacientes, registrado no array 'diet', com os intervalos calóricos definidos no seu plano de refeições, 'mealPlan', durante um determinado período. Os IDs dos pacientes cuja ingestão calórica esteja fora do intervalo são armazenados e contados.

Parâmetros

diet	Ponteiro para o array de estruturas 'Diet', que contém os dados de consumo de calorias dos pacientes.
mealPlan	Ponteiro para o array de estruturas 'MealPlan', que define os intervalos calóricos para os pacientes.
period	Estrutura 'Period' que define o período de tempo durante o qual o consumo é avaliado.
max_size	Tamanho máximo dos arrays 'diet' e 'mealPlan'.

Retorna

Retorna o número de pacientes cujo consumo de calorias está fora do intervalo estipulado no seu plano de refeições.

Nota

Esta função pressupõe que os arrays 'diet' e 'mealPlan' são válidos, e que o tamanho máximo dos arrays é respeitado. Além disso, assume-se que os IDs dos pacientes são únicos.

Aviso

A função utiliza um array fixo de tamanho 100 para armazenar os IDs dos pacientes. Se houver mais de 100 pacientes únicos no período especificado, a função não poderá rastrear todos e os resultados podem não ser completos.

Referências dateInPeriod(), Diet::ID e sortDescending().

Referenciado por handleOutOfRange().

4.2.2.5 printTable()

```
void printTable (
            MealPlan * mealPlans,
            Diet * diets,
            Patients * patients,
            int max_size )
```

Imprime uma tabela com o resumo dos planos alimentares e o consumo calórico dos pacientes.

Esta função constrói e imprime uma tabela detalhada que mostra o plano alimentar de cada paciente, incluindo os tipos de refeição, o período de cada plano, as calorias mínimas e máximas estipuladas, e o total de calorias consumidas. A função itera sobre os arrays 'mealPlans' e 'diets', preenchendo uma estrutura auxiliar 'infoTable' para armazenar as informações consolidadas antes de imprimir.

Parâmetros

mealPlans	Ponteiro para o array de estruturas 'MealPlan', representando os planos alimentares.
diets	Ponteiro para o array de estruturas 'Diet', representando o consumo de calorias dos pacientes.
patients	Ponteiro para o array de estruturas 'Patients', contendo informações sobre os pacientes.
max_size	Tamanho máximo dos arrays 'mealPlans', 'diets' e 'patients'.

Nota

Esta função pressupõe que os arrays 'mealPlans', 'diets' e 'patients' são válidos e que o tamanho máximo dos arrays é respeitado. Além disso, assume-se que os IDs dos pacientes são únicos.

Aviso

A função utiliza um array fixo 'infoTable' de tamanho 100 para armazenar as informações consolidadas. Se houver mais de 100 pacientes únicos, a tabela não poderá representar todos eles.

Referências Period::begin, Diet::calories, InfoTable::calories, MealPlan::date, dateInPeriod(), Date::day, Period::end, Patients::ID, Diet::ID, MealPlan::ID, MealPlan::maxCal, InfoTable::maxCal, InfoTable::meal, MealPlan::minCal, InfoTable::minCal, Date::month, Patients::name, InfoTable::patient, InfoTable::period e Date::year.

Referenciado por handlePrintTable().

4.3 logic.h

Ir para a documentação deste ficheiro.

```
1 #ifndef LOGIC_H
2 #define LOGIC_H
3
4 #include "types.h"
5
27 //Funcoes
28
48 void printTable(MealPlan *mealPlans, Diet *diets, Patients *patients, int max_size);
49
71 int exceededCalories(Diet *diet,int max_size, int calories, Period period);
72
93 int outOfRange(Diet *diet, MealPlan *mealPlan, Period period, int max_size);
94
117 int listMealPlan(MealPlan *mealPlan, Period period, int max_size, char *mealType, int IDNum);
118
141 float averageCalories(Diet *diet, Period period, int max_size, char *mealType, int IDNum);
142
143 #endif // LOGIC_H
```

4.4 Referência ao ficheiro src/main.c

Ponto de entrada principal do programa e interface de utilizador.

```
#include "utils.h"
#include "logic.h"
#include "types.h"
#include "menu.h"
#include <string.h>
#include <stdio.h>
```

Macros

- #define MAX_SIZE 100

Funções

- int main ()

4.4.1 Descrição detalhada

Ponto de entrada principal do programa e interface de utilizador.

Este ficheiro implementa a função 'main', que serve como o ponto de entrada principal do programa. Aqui é apresentada a interface de utilizador no formato de um menu de texto, permitindo ao utilizador interagir com o programa e aceder a diferentes funcionalidades relacionadas com a gestão de dietas e planos alimentares. O programa oferece opções para calcular calorias excedidas, verificar refeições fora do intervalo calórico, listar planos nutricionais, calcular a média de calorias consumidas e visualizar uma tabela de informações.

As operações implementadas neste ficheiro fazem uso intensivo das funções definidas em 'utils.h' e 'logic.h', e dos tipos de dados definidos em 'types.h'. Dados iniciais são carregados de ficheiros de texto, e o utilizador pode interagir com estes dados através de várias opções de menu.

Nota

Este ficheiro depende das definições em 'utils.h', 'logic.h' e 'types.h' para a sua funcionalidade.

Aviso

A função 'main' assume que os ficheiros de dados necessários estão disponíveis e no formato correto. O programa pode não funcionar como esperado se estes ficheiros estiverem ausentes ou malformados.

4.4.2 Documentação das macros

4.4.2.1 MAX_SIZE

```
#define MAX_SIZE 100
```

4.4.3 Documentação das funções

4.4.3.1 main()

```
int main ( )
```

Referências DIET, handleAverageCalories(), handleExceededCalories(), handleMealPlan(), handleOutOfRange(), handlePrintTable(), initializeDiets(), initializeMealPlans(), initializePatients(), MAX_SIZE, MEAL_PLAN, PATIENTS, readFile(), showMenuAndGetChoice() e waitForUserInput().

4.5 Referência ao ficheiro src/menu.c

Funções de Interface do Menu.

```
#include "utils.h"
#include "menu.h"
#include "logic.h"
#include "types.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Funções

- void initializeDiets (Diet diets[ ], int size)

  Inicializa um array de Diet com valores padrão.
- void initializePatients (Patients patients[ ], int size)

  Inicializa um array de Patients com valores padrão.
- void initializeMealPlans (MealPlan mealPlans[ ], int size)

  Inicializa um array de MealPlan com valores padrão.
- void handleExceededCalories (Diet diets[ ], int size)

  Processa e exibe o número de pacientes que excederam um limite de calorias.
- void handleOutOfRange (Diet diets[ ], MealPlan mealPlans[ ], int numDiets)

  Identifica e exibe refeições que estão fora do intervalo calórico estabelecido.
- void handleMealPlan (MealPlan mealPlans[ ], int size)

  Gerencia e exibe um plano de refeições para um paciente específico.
- void handleAverageCalories (Diet diets[ ], int numDiets)

  Calcula e exibe a média de calorias consumidas por um paciente.
- void handlePrintTable (MealPlan mealPlans[ ], Diet diets[ ], Patients patients[ ], int size)

  Exibe uma tabela com informações consolidadas de dietas e planos de refeições.
- int showMenuAndGetChoice ()

  Mostra o menu de opções e obtém a escolha do utilizador.
- void clearScreen ()

  Limpa o ecra do terminal. Verifica o sistema operativo para executar o comando correto.
- void waitForUserInput ()

  Aguarda até que o utilizador pressione Enter.

4.5.1 Descrição detalhada

Funções de Interface do Menu.

Este ficheiro contém as definições das funções usadas para criar e gerir a interface do menu do programa. Inclui funções para inicializar dados, manipular opções de menu, e interações de utilizador, como a exibição do menu, limpeza do ecra e espera de entrada do utilizador. Estas funções são projetadas para facilitar a navegação e interação do utilizador com as diversas funcionalidades do sistema.

4.5.2 Documentação das funções

4.5.2.1 clearScreen()

```
void clearScreen ( )
```

Limpa o ecra do terminal. Verifica o sistema operativo para executar o comando correto.

Referenciado por showMenuAndGetChoice().

4.5.2.2 handleAverageCalories()

```
void handleAverageCalories (
            Diet diets[],
            int numDiets )
```

Calcula e exibe a média de calorias consumidas por um paciente.

Parâmetros

diets	Array de Diet.
numDiets	Número de elementos no array de Diet.

Referências averageCalories(), clearInputBuffer() e fillPeriod().

Referenciado por main().

4.5.2.3 handleExceededCalories()

```
void handleExceededCalories (
            Diet diets[],
            int size )
```

Processa e exibe o número de pacientes que excederam um limite de calorias.

Parâmetros

diets	Array de Diet contendo informações dietéticas.
size	Tamanho do array de Diet.

Referências exceededCalories() e fillPeriod().

Referenciado por main().

4.5.2.4 handleMealPlan()

```
void handleMealPlan (
            MealPlan mealPlans[],
            int size )
```

Gerencia e exibe um plano de refeições para um paciente específico.

Parâmetros

mealPlans	Array de MealPlan.
size	Tamanho do array de MealPlan.

Referências fillPeriod() e listMealPlan().

Referenciado por main().

4.5.2.5 handleOutOfRange()

```
void handleOutOfRange (
            Diet diets[],
            MealPlan mealPlans[],
            int numDiets )
```

Identifica e exibe refeições que estão fora do intervalo calórico estabelecido.

Parâmetros

diets	Array de Diet contendo informações dietéticas.
mealPlans	Array de MealPlan.
numDiets	Número de elementos no array de Diet.

Referências fillPeriod() e outOfRange().

Referenciado por main().

4.5.2.6 handlePrintTable()

```
void handlePrintTable (
            MealPlan mealPlans[],
            Diet diets[],
            Patients patients[],
            int size )
```

Exibe uma tabela com informações consolidadas de dietas e planos de refeições.

Parâmetros

mealPlans	Array de MealPlan.
diets	Array de Diet.
patients	Array de Patients.
size	Tamanho dos arrays.

Referências printTable().

Referenciado por main().

4.5.2.7 initializeDiets()

```
void initializeDiets (
            Diet diets[],
            int size )
```

Inicializa um array de Diet com valores padrão.

Parâmetros

diets	Array de estruturas Diet a ser inicializado.
size	Tamanho do array a ser inicializado.

Referências Diet::ID.

Referenciado por main().

4.5.2.8 initializeMealPlans()

```
void initializeMealPlans (
            MealPlan mealPlans[],
            int size )
```

Inicializa um array de MealPlan com valores padrão.

Parâmetros

mealPlans	Array de estruturas MealPlan a ser inicializado.
size	Tamanho do array a ser inicializado.

Referências MealPlan::ID.

Referenciado por main().

4.5.2.9 initializePatients()

```
void initializePatients (
            Patients patients[],
            int size )
```

Inicializa um array de Patients com valores padrão.

Parâmetros

patients	Array de estruturas Patients a ser inicializado.
size	Tamanho do array a ser inicializado.

Referências Patients::ID.

Referenciado por main().

4.5.2.10 showMenuAndGetChoice()

```
int showMenuAndGetChoice ( )
```

Mostra o menu de opções e obtém a escolha do utilizador.

Retorna

int A escolha do utilizador.

Referências clearInputBuffer() e clearScreen().

Referenciado por main().

4.5.2.11 waitForUserInput()

```
void waitForUserInput ( )
```

Aguarda até que o utilizador pressione Enter.

Referenciado por main().

4.6 Referência ao ficheiro src/menu.h

Cabeçalho para as Funções de Interface do Menu.

```
#include "types.h"
#include "utils.h"
#include "logic.h"
```

Funções

- void initializeDiets (Diet diets[ ], int size)

  Inicializa um array de Diet com valores padrão.
- void initializePatients (Patients patients[ ], int size)

  Inicializa um array de Patients com valores padrão.
- void initializeMealPlans (MealPlan mealPlans[ ], int size)

  Inicializa um array de MealPlan com valores padrão.
- void handleExceededCalories (Diet diets[ ], int size)

  Processa e exibe o número de pacientes que excederam um limite de calorias.
- void handleOutOfRange (Diet diets[ ], MealPlan mealPlans[ ], int numDiets)

  Identifica e exibe refeições que estão fora do intervalo calórico estabelecido.
- void handleMealPlan (MealPlan mealPlans[ ], int size)

  Gerencia e exibe um plano de refeições para um paciente específico.
- void handleAverageCalories (Diet diets[ ], int numDiets)

  Calcula e exibe a média de calorias consumidas por um paciente.
- void handlePrintTable (MealPlan mealPlans[ ], Diet diets[ ], Patients patients[ ], int size)

  Exibe uma tabela com informações consolidadas de dietas e planos de refeições.
- void clearScreen ()

  Limpa o ecra do terminal. Verifica o sistema operativo para executar o comando correto.
- void waitForUserInput ()

  Aguarda até que o utilizador pressione Enter.
- int showMenuAndGetChoice ()

  Mostra o menu de opções e obtém a escolha do utilizador.

4.6.1 Descrição detalhada

Cabeçalho para as Funções de Interface do Menu.

Este ficheiro de cabeçalho declara as funções utilizadas na interface do menu do programa. Contém os protótipos das funções definidas em menu.c, abrangendo a inicialização de dados e a gestão das opções do menu. Este ficheiro promove a modularidade e a manutenção do código, facilitando a integração e reutilização das funções em diferentes partes do programa.

4.6.2 Documentação das funções

4.6.2.1 clearScreen()

```
void clearScreen ( )
```

Limpa o ecra do terminal. Verifica o sistema operativo para executar o comando correto.

Referenciado por showMenuAndGetChoice().

4.6.2.2 handleAverageCalories()

```
void handleAverageCalories (
            Diet diets[],
            int numDiets )
```

Calcula e exibe a média de calorias consumidas por um paciente.

Parâmetros

diets	Array de Diet.
numDiets	Número de elementos no array de Diet.

Referências averageCalories(), clearInputBuffer() e fillPeriod().

Referenciado por main().

4.6.2.3 handleExceededCalories()

```
void handleExceededCalories (
            Diet diets[],
            int size )
```

Processa e exibe o número de pacientes que excederam um limite de calorias.

Parâmetros

diets	Array de Diet contendo informações dietéticas.
size	Tamanho do array de Diet.

Referências exceededCalories() e fillPeriod().

Referenciado por main().

4.6.2.4 handleMealPlan()

```
void handleMealPlan (
            MealPlan mealPlans[],
            int size )
```

Gerencia e exibe um plano de refeições para um paciente específico.

Parâmetros

mealPlans	Array de MealPlan.
size	Tamanho do array de MealPlan.

Referências fillPeriod() e listMealPlan().

Referenciado por main().

4.6.2.5 handleOutOfRange()

```
void handleOutOfRange (
            Diet diets[],
            MealPlan mealPlans[],
            int numDiets )
```

Identifica e exibe refeições que estão fora do intervalo calórico estabelecido.

Parâmetros

diets	Array de Diet contendo informações dietéticas.
mealPlans	Array de MealPlan.
numDiets	Número de elementos no array de Diet.

Referências fillPeriod() e outOfRange().

Referenciado por main().

4.6.2.6 handlePrintTable()

```
void handlePrintTable (
            MealPlan mealPlans[],
            Diet diets[],
            Patients patients[],
            int size )
```

Exibe uma tabela com informações consolidadas de dietas e planos de refeições.

Parâmetros

mealPlans	Array de MealPlan.
diets	Array de Diet.
patients	Array de Patients.
size	Tamanho dos arrays.

Referências printTable().

Referenciado por main().

4.6.2.7 initializeDiets()

```
void initializeDiets (
            Diet diets[],
            int size )
```

Inicializa um array de Diet com valores padrão.

Parâmetros

diets	Array de estruturas Diet a ser inicializado.
size	Tamanho do array a ser inicializado.

Referências Diet::ID.

Referenciado por main().

4.6.2.8 initializeMealPlans()

```
void initializeMealPlans (
            MealPlan mealPlans[],
            int size )
```

Inicializa um array de MealPlan com valores padrão.

Parâmetros

mealPlans	Array de estruturas MealPlan a ser inicializado.
size	Tamanho do array a ser inicializado.

Referências MealPlan::ID.

Referenciado por main().

4.6.2.9 initializePatients()

```
void initializePatients (
            Patients patients[],
            int size )
```

Inicializa um array de Patients com valores padrão.

Parâmetros

patients	Array de estruturas Patients a ser inicializado.
size	Tamanho do array a ser inicializado.

Referências Patients::ID.

Referenciado por main().

4.6.2.10 showMenuAndGetChoice()

```
int showMenuAndGetChoice ( )
```

Mostra o menu de opções e obtém a escolha do utilizador.

Retorna

int A escolha do utilizador.

Referências clearInputBuffer() e clearScreen().

Referenciado por main().

4.6.2.11 waitForUserInput()

```
void waitForUserInput ( )
```

Aguarda até que o utilizador pressione Enter.

Referenciado por main().

4.7 menu.h

Ir para a documentação deste ficheiro.

```
1 #ifndef MENU_H
2 #define MENU_H
3
4 #include "types.h"
5 #include "utils.h"
6 #include "logic.h"
7
19 void initializeDiets(Diet diets[], int size);
20 void initializePatients(Patients patients[], int size);
21 void initializeMealPlans(MealPlan mealPlans[], int size);
22 void handleExceededCalories(Diet diets[], int size);
23 void handleOutOfRange(Diet diets[], MealPlan mealPlans[], int numDiets);
24 void handleMealPlan(MealPlan mealPlans[], int size);
25 void handleAverageCalories(Diet diets[], int numDiets);
26 void handlePrintTable(MealPlan mealPlans[], Diet diets[], Patients patients[], int size);
27 void clearScreen();
28 void waitForUserInput();
29 int showMenuAndGetChoice();
30
31 #endif // MENU_H
```

4.8 Referência ao ficheiro src/types.h

Definição de estruturas de dados e tipos enumerados para o programa.

Estruturas de Dados

- struct Date

  Estrutura para representar uma data.
- struct Period

  Estrutura para representar um período de tempo definido por uma data de início e uma data de fim.
- struct Patients

  Estrutura para representar as informações de um paciente.
- struct Diet

  Estrutura para representar informações sobre a dieta de um paciente.
- struct IDCalories

  Estrutura para representar o consumo de calorias associado a um identificador de paciente.
- struct MealPlan

  Estrutura para representar um plano alimentar para um paciente.
- struct InfoTable

  Estrutura para representar informações detalhadas de um plano alimentar e o consumo real de um paciente.

Enumerações

- enum FileType { PATIENTS , DIET , MEAL_PLAN }

  Enumeração para representar diferentes tipos de ficheiros utilizados no sistema.

4.8.1 Descrição detalhada

Definição de estruturas de dados e tipos enumerados para o programa.

Este ficheiro de cabeçalho contém as definições das principais estruturas de dados e tipos enumerados utilizados no programa. Inclui estruturas para representar datas, períodos, informações de pacientes, detalhes de dietas, planos de refeições e outras informações relevantes. Este ficheiro é a base para a manipulação de dados em todo o programa, fornecendo os blocos de construção essenciais para as operações de lógica e utilidade.

As estruturas e tipos enumerados definidos incluem:

- 'Date': Representa uma data.
- 'Period': Define um período com datas de início e fim.
- 'Patients': Armazena informações sobre pacientes.
- 'Diet': Detalha uma dieta, incluindo a ingestão calórica.
- 'IDCalories': Associa um ID a um valor calórico.
- 'MealPlan': Define um plano de refeições com limites calóricos.
- 'InfoTable': Estrutura para armazenar e apresentar informações consolidadas.
- 'FileType': Enumeração dos tipos de ficheiros para operações de leitura de dados.

Nota

Estas estruturas e tipos enumerados são fundamentais para a estrutura de dados do programa e são amplamente utilizados nas diversas funções e operações implementadas.

4.8.2 Documentação dos valores da enumeração

4.8.2.1 FileType

```
enum FileType
```

Enumeração para representar diferentes tipos de ficheiros utilizados no sistema.

Esta enumeração é usada para distinguir entre diferentes tipos de ficheiros de dados que podem ser lidos e processados pelo programa. Facilita a manipulação e o processamento de diferentes conjuntos de dados, permitindo que o programa identifique e trate adequadamente cada tipo de ficheiro.

Valores de enumerações

PATIENTS	Representa um ficheiro de dados contendo informações sobre pacientes.
DIET	Representa um ficheiro de dados contendo informações sobre as dietas dos pacientes.
MEAL_PLAN	Representa um ficheiro de dados contendo informações sobre os planos alimentares dos pacientes.

4.9 types.h

Ir para a documentação deste ficheiro.

```
1 #ifndef TYPES_H
2 #define TYPES_H
3
4
28 //DataTypes
29
47 typedef struct {
48         int day;
49         int month;
50         int year;
51 } Date;
52
70 typedef struct {
71         Date begin;
72         Date end;
73 } Period;
74
93 typedef struct {
94         int ID;
95         char name[50];
96         int phoneNumber;
97 } Patients;
98
124 typedef struct {
125         int ID;
126         Date date;
127         char meal[50];
128         char food[50];
129         int calories;
130 } Diet;
131
148 typedef struct {
149         int ID;
150         int calories;
151 } IDCalories;
152
178 typedef struct {
179         int ID;
180         Date date;
181         char meal[50];
182         int minCal;
```

```
183         int maxCal;
184 } MealPlan;
185
213 typedef struct {
214         Patients patient;
215         char meal[50];
216         Period period;
217         int minCal;
218         int maxCal;
219         int calories;
220 } InfoTable;
221
240 typedef enum {
241         PATIENTS,
242         DIET,
243         MEAL_PLAN
244 } FileType;
245
246 #endif // TYPES_H
```

4.10 Referência ao ficheiro src/utils.c

Implementação das funções de utilidade para o programa.

```
#include "utils.h"
#include <stdio.h>
#include <stdlib.h>
```

Funções

- int readFile (char *path, void *data, int max_size, FileType fileType)

  Lê dados de um ficheiro e armazena-os numa estrutura de dados especificada.
- int dateInPeriod (Date date, Period period)

  Verifica se uma data específica está dentro de um determinado período.
- void printDate (Date date)

  Imprime uma data no formato padrão DD-MM-AAAA.
- void printPeriod (Period period)

  Imprime um período, mostrando as datas de início e fim.
- void clearInputBuffer ()

  Limpa o buffer de entrada do standard input (stdin).
- void sortDescending (int ids[ ], int numberEl)

  Ordena um array de inteiros em ordem decrescente.
- void fillPeriod (Period *period)

  Preenche um período com datas de início e fim.

4.10.1 Descrição detalhada

Implementação das funções de utilidade para o programa.

Este ficheiro contém as implementações das funções de utilidade declaradas em 'utils.h'. As funções aqui presentes oferecem uma variedade de operações auxiliares, incluindo leitura de dados de ficheiros, manipulação e impressão de datas e períodos, limpeza do buffer de entrada e ordenação de arrays. Estas funções são utilizadas em várias partes do programa para realizar tarefas comuns, como processamento de dados de entrada e apresentação de informações de forma legível.

As funções implementadas neste ficheiro incluem:

- Leitura e interpretação de dados de ficheiros com formatos específicos.
- Verificação se uma data está dentro de um período especificado.
- Impressão formatada de datas e períodos.
- Limpeza do buffer de entrada para evitar leituras indesejadas de dados.
- Ordenação de arrays de inteiros em ordem decrescente.

Nota

As funções neste ficheiro são dependentes das estruturas de dados e tipos enumerados definidos em 'types.h' e declarados em 'utils.h'.

4.10.2 Documentação das funções

4.10.2.1 clearInputBuffer()

```
void clearInputBuffer ( )
```

Limpa o buffer de entrada do standard input (stdin).

Esta função remove todos os caracteres remanescentes no buffer de entrada do stdin até encontrar um caractere de nova linha ('
') ou o fim do ficheiro (EOF). É particularmente útil após a leitura de dados de entrada para garantir que nenhuma entrada indesejada permaneça no buffer e afete leituras futuras.

Nota

Esta função deve ser utilizada com cuidado, especialmente em ambientes onde o buffer de entrada pode conter dados relevantes que não devem ser descartados.

Aviso

A função pode bloquear a execução se não houver dados no buffer de entrada e não for atingido o fim do ficheiro, especialmente em ambientes não interativos.

Referenciado por fillPeriod(), handleAverageCalories() e showMenuAndGetChoice().

4.10.2.2 dateInPeriod()

```
int dateInPeriod (
            Date date,
            Period period )
```

Verifica se uma data específica está dentro de um determinado período.

Esta função avalia se uma data fornecida está antes, dentro ou depois de um período especificado. O período é definido por uma data de início e uma data de fim. A função compara a data fornecida com as datas de início e fim do período, determinando a sua relação temporal com o mesmo.

Parâmetros

date	Estrutura 'Date' representando a data a ser verificada.
period	Estrutura 'Period' representando o período contra o qual a data será comparada.

Retorna

Retorna -1 se a data estiver antes do período, 1 se estiver dentro do período, e 2 se estiver após o período.

Nota

Esta função assume que as estruturas 'Date' e 'Period' são válidas e que as datas estão no formato correto.

Aviso

A função não verifica a coerência do período em si (por exemplo, se a data de início é anterior à data de fim).

Referências Period::begin, Date::day, Period::end, Date::month e Date::year.

Referenciado por averageCalories(), exceededCalories(), listMealPlan(), outOfRange() e printTable().

4.10.2.3 fillPeriod()

```
void fillPeriod (
            Period * period )
```

Preenche um período com datas de início e fim.

Esta função solicita ao utilizador que introduza as datas de início e fim de um período. As datas são lidas do standard input (teclado) e armazenadas na estrutura de período fornecida.

Parâmetros

period	Ponteiro para a estrutura de período que será preenchida. Espera-se que este ponteiro seja válido e não seja NULL.

Aviso

A função usa `scanf` para leitura de dados, o que pode levar a erros de entrada se o formato esperado não for respeitado.

Exemplo de uso:
```
Period meuPeriodo;
fillPeriod(&meuPeriodo);
```

Referências Period::begin, clearInputBuffer(), Date::day, Period::end, Date::month e Date::year.

Referenciado por handleAverageCalories(), handleExceededCalories(), handleMealPlan() e handleOutOfRange().

4.10.2.4 printDate()

```
void printDate (
            Date date )
```

Imprime uma data no formato padrão DD-MM-AAAA.

Esta função recebe uma estrutura 'Date' e imprime-a no standard output (stdout) no formato dia-mês-ano (DD-↩ MM-AAAA), onde DD é o dia, MM é o mês e AAAA é o ano. A função é útil para exibir datas de forma legível e padronizada.

Parâmetros

date	Estrutura 'Date' que contém a data a ser impressa.

Nota

Esta função assume que a estrutura 'Date' fornecida é válida e que os valores de dia, mês e ano estão dentro de intervalos aceitáveis para representar uma data válida.

Aviso

A função não realiza validação da data fornecida; portanto, a validade da data (como dias corretos para determinados meses, anos bissextos, etc.) deve ser garantida antes de chamar esta função.

Referências Date::day, Date::month e Date::year.

Referenciado por printPeriod().

4.10.2.5 printPeriod()

```
void printPeriod (
            Period period )
```

Imprime um período, mostrando as datas de início e fim.

Esta função recebe uma estrutura 'Period' e imprime as datas de início e fim do período no standard output (stdout). Utiliza a função 'printDate' para imprimir as datas de início e fim no formato padrão. É útil para exibir períodos de forma legível e clara, mostrando claramente a duração de um intervalo de tempo.

Parâmetros

period	Estrutura 'Period' que contém as datas de início e fim a serem impressas.

Nota

Esta função assume que as datas de início e fim dentro da estrutura 'Period' são válidas e que a data de início precede a data de fim.

Aviso

A função não realiza validação das datas dentro do período; portanto, a coerência do período (como a data de início sendo anterior à data de fim) deve ser garantida antes de chamar esta função.

Referências Period::begin, Period::end e printDate().

4.10.2.6 readFile()

```
int readFile (
              char * path,
              void * data,
              int max_size,
              FileType fileType )
```

Lê dados de um ficheiro e armazena-os numa estrutura de dados especificada.

Esta função abre um ficheiro no caminho especificado e lê os seus conteúdos linha por linha, armazenando os dados numa estrutura de acordo com o tipo de ficheiro fornecido. As estruturas suportadas são 'Patients', 'Diet' e 'MealPlan'. A função é capaz de interpretar diferentes formatos de dados com base no tipo de ficheiro fornecido.

Parâmetros

path	Caminho para o ficheiro a ser lido.
data	Ponteiro para a estrutura de dados onde os dados lidos serão armazenados.
max_size	Número máximo de elementos a serem lidos e armazenados na estrutura de dados.
fileType	Enumeração 'FileType' que indica o tipo de dados esperado no ficheiro (ex: PATIENTS, DIET, MEAL_PLAN).

Retorna

Retorna o número de elementos lidos e armazenados com sucesso na estrutura de dados. Retorna 0 se não for possível abrir o ficheiro ou se o tipo de ficheiro não for suportado.

Nota

A função pressupõe que o ponteiro 'data' é válido e que o espaço de memória suficiente foi alocado para armazenar os dados.

Aviso

Esta função não realiza verificações extensivas de validade dos dados lidos do ficheiro. A validade e consistência dos dados no ficheiro são assumidas como corretas.

Referências Diet::calories, Diet::date, MealPlan::date, DIET, Diet::food, MealPlan::maxCal, Diet::meal, MealPlan::meal, MEAL_PLAN, MealPlan::minCal, Date::month, PATIENTS e Date::year.

Referenciado por main().

4.10.2.7 sortDescending()

```
void sortDescending (
            int ids[],
            int numberEl )
```

Ordena um array de inteiros em ordem decrescente.

Esta função realiza a ordenação de um array de inteiros utilizando o algoritmo de ordenação 'bubble sort' modificado. A ordenação é feita em ordem decrescente, onde o maior elemento é colocado no início do array e o menor no final. O algoritmo compara pares de elementos adjacentes e os troca de lugar se estiverem na ordem errada, continuando este processo até que todo o array esteja ordenado.

Parâmetros

ids	Array de inteiros que será ordenado.
numberEl	Número de elementos no array 'ids'.

Nota

A função modifica o array 'ids' diretamente. Portanto, o array original será alterado para refletir a nova ordem dos elementos.

Aviso

A função assume que o tamanho do array 'ids' é pelo menos igual a 'numberEl'. Passar um valor de 'numberEl' maior do que o tamanho real do array pode resultar em comportamento indefinido.

Referenciado por outOfRange().

4.11 Referência ao ficheiro src/utils.h

Cabeçalho das funções de utilidade para o programa.

```
#include "types.h"
```

Funções

- int readFile (char *path, void *data, int max_size, FileType fileType)
  Lê dados de um ficheiro e armazena-os numa estrutura de dados especificada.
- int dateInPeriod (Date date, Period period)
  Verifica se uma data específica está dentro de um determinado período.
- void sortDescending (int ids[ ], int numberEl)
  Ordena um array de inteiros em ordem decrescente.
- void printDate (Date date)
  Imprime uma data no formato padrão DD-MM-AAAA.
- void printPeriod (Period period)
  Imprime um período, mostrando as datas de início e fim.
- void clearInputBuffer ()
  Limpa o buffer de entrada do standard input (stdin).
- void fillPeriod (Period *period)
  Preenche um período com datas de início e fim.

4.11.1 Descrição detalhada

Cabeçalho das funções de utilidade para o programa.

Este ficheiro de cabeçalho contém as declarações das funções de utilidade usadas em várias partes do programa. Inclui funções para leitura de ficheiros, manipulação de datas, ordenação de arrays, impressão de datas e períodos, e limpeza do buffer de entrada. As funções aqui definidas são auxiliares à lógica principal do programa e são utilizadas para realizar operações comuns de forma eficiente.

Nota

Este ficheiro inclui 'types.h', que contém as definições das estruturas de dados utilizadas nas funções declaradas.

As funções implementadas oferecem operações como:

- Leitura de dados de ficheiros com formatos específicos.
- Verificação se uma data está dentro de um período especificado.
- Ordenação de arrays de inteiros em ordem decrescente.
- Impressão formatada de datas e períodos.
- Limpeza do buffer de entrada para evitar leituras indesejadas.

4.11.2 Documentação das funções

4.11.2.1 clearInputBuffer()

```
void clearInputBuffer ( )
```

Limpa o buffer de entrada do standard input (stdin).

Esta função remove todos os caracteres remanescentes no buffer de entrada do stdin até encontrar um caractere de nova linha ('
') ou o fim do ficheiro (EOF). É particularmente útil após a leitura de dados de entrada para garantir que nenhuma entrada indesejada permaneça no buffer e afete leituras futuras.

Nota

Esta função deve ser utilizada com cuidado, especialmente em ambientes onde o buffer de entrada pode conter dados relevantes que não devem ser descartados.

Aviso

A função pode bloquear a execução se não houver dados no buffer de entrada e não for atingido o fim do ficheiro, especialmente em ambientes não interativos.

Referenciado por fillPeriod(), handleAverageCalories() e showMenuAndGetChoice().

4.11.2.2 dateInPeriod()

```
int dateInPeriod (
            Date date,
            Period period )
```

Verifica se uma data específica está dentro de um determinado período.

Esta função avalia se uma data fornecida está antes, dentro ou depois de um período especificado. O período é definido por uma data de início e uma data de fim. A função compara a data fornecida com as datas de início e fim do período, determinando a sua relação temporal com o mesmo.

Parâmetros

date	Estrutura 'Date' representando a data a ser verificada.
period	Estrutura 'Period' representando o período contra o qual a data será comparada.

Retorna

Retorna -1 se a data estiver antes do período, 1 se estiver dentro do período, e 2 se estiver após o período.

Nota

Esta função assume que as estruturas 'Date' e 'Period' são válidas e que as datas estão no formato correto.

Aviso

A função não verifica a coerência do período em si (por exemplo, se a data de início é anterior à data de fim).

Referências Period::begin, Date::day, Period::end, Date::month e Date::year.

Referenciado por averageCalories(), exceededCalories(), listMealPlan(), outOfRange() e printTable().

4.11.2.3 fillPeriod()

```
void fillPeriod (
            Period * period )
```

Preenche um período com datas de início e fim.

Esta função solicita ao utilizador que introduza as datas de início e fim de um período. As datas são lidas do standard input (teclado) e armazenadas na estrutura de período fornecida.

Parâmetros

period	Ponteiro para a estrutura de período que será preenchida. Espera-se que este ponteiro seja válido e não seja NULL.

Aviso

A função usa `scanf` para leitura de dados, o que pode levar a erros de entrada se o formato esperado não for respeitado.

Exemplo de uso:
```
Period meuPeriodo;
fillPeriod(&meuPeriodo);
```

Referências Period::begin, clearInputBuffer(), Date::day, Period::end, Date::month e Date::year.

Referenciado por handleAverageCalories(), handleExceededCalories(), handleMealPlan() e handleOutOfRange().

4.11.2.4 printDate()

```
void printDate (
            Date date )
```

Imprime uma data no formato padrão DD-MM-AAAA.

Esta função recebe uma estrutura 'Date' e imprime-a no standard output (stdout) no formato dia-mês-ano (DD-↩ MM-AAAA), onde DD é o dia, MM é o mês e AAAA é o ano. A função é útil para exibir datas de forma legível e padronizada.

Parâmetros

date	Estrutura 'Date' que contém a data a ser impressa.

Nota

Esta função assume que a estrutura 'Date' fornecida é válida e que os valores de dia, mês e ano estão dentro de intervalos aceitáveis para representar uma data válida.

Aviso

A função não realiza validação da data fornecida; portanto, a validade da data (como dias corretos para determinados meses, anos bissextos, etc.) deve ser garantida antes de chamar esta função.

Referências Date::day, Date::month e Date::year.

Referenciado por printPeriod().

4.11.2.5 printPeriod()

```
void printPeriod (
            Period period )
```

Imprime um período, mostrando as datas de início e fim.

Esta função recebe uma estrutura 'Period' e imprime as datas de início e fim do período no standard output (stdout). Utiliza a função 'printDate' para imprimir as datas de início e fim no formato padrão. É útil para exibir períodos de forma legível e clara, mostrando claramente a duração de um intervalo de tempo.

Parâmetros

period	Estrutura 'Period' que contém as datas de início e fim a serem impressas.

Nota

Esta função assume que as datas de início e fim dentro da estrutura 'Period' são válidas e que a data de início precede a data de fim.

Aviso

A função não realiza validação das datas dentro do período; portanto, a coerência do período (como a data de início sendo anterior à data de fim) deve ser garantida antes de chamar esta função.

Referências Period::begin, Period::end e printDate().

4.11.2.6 readFile()

```
int readFile (
              char * path,
              void * data,
              int max_size,
              FileType fileType )
```

Lê dados de um ficheiro e armazena-os numa estrutura de dados especificada.

Esta função abre um ficheiro no caminho especificado e lê os seus conteúdos linha por linha, armazenando os dados numa estrutura de acordo com o tipo de ficheiro fornecido. As estruturas suportadas são 'Patients', 'Diet' e 'MealPlan'. A função é capaz de interpretar diferentes formatos de dados com base no tipo de ficheiro fornecido.

Parâmetros

path	Caminho para o ficheiro a ser lido.
data	Ponteiro para a estrutura de dados onde os dados lidos serão armazenados.
max_size	Número máximo de elementos a serem lidos e armazenados na estrutura de dados.
fileType	Enumeração 'FileType' que indica o tipo de dados esperado no ficheiro (ex: PATIENTS, DIET, MEAL_PLAN).

Retorna

Retorna o número de elementos lidos e armazenados com sucesso na estrutura de dados. Retorna 0 se não for possível abrir o ficheiro ou se o tipo de ficheiro não for suportado.

Nota

A função pressupõe que o ponteiro 'data' é válido e que o espaço de memória suficiente foi alocado para armazenar os dados.

Aviso

Esta função não realiza verificações extensivas de validade dos dados lidos do ficheiro. A validade e consistência dos dados no ficheiro são assumidas como corretas.

Referências Diet::calories, Diet::date, MealPlan::date, DIET, Diet::food, MealPlan::maxCal, Diet::meal, MealPlan::meal, MEAL_PLAN, MealPlan::minCal, Date::month, PATIENTS e Date::year.

Referenciado por main().

4.11.2.7 sortDescending()

```
void sortDescending (
            int ids[],
            int numberEl )
```

Ordena um array de inteiros em ordem decrescente.

Esta função realiza a ordenação de um array de inteiros utilizando o algoritmo de ordenação 'bubble sort' modificado. A ordenação é feita em ordem decrescente, onde o maior elemento é colocado no início do array e o menor no final. O algoritmo compara pares de elementos adjacentes e os troca de lugar se estiverem na ordem errada, continuando este processo até que todo o array esteja ordenado.

Parâmetros

ids	Array de inteiros que será ordenado.
numberEl	Número de elementos no array 'ids'.

Nota

A função modifica o array 'ids' diretamente. Portanto, o array original será alterado para refletir a nova ordem dos elementos.

Aviso

A função assume que o tamanho do array 'ids' é pelo menos igual a 'numberEl'. Passar um valor de 'numberEl' maior do que o tamanho real do array pode resultar em comportamento indefinido.

Referenciado por outOfRange().

4.12 utils.h

Ir para a documentação deste ficheiro.

```
1 #ifndef UTILS_H
2 #define UTILS_H
3
4 #include "types.h"
5
48 int readFile(char *path, void *data, int max_size, FileType fileType);
49
66 int dateInPeriod(Date date, Period period);
67
85 void sortDescending(int ids[], int numberEl);
86
102 void printDate(Date date);
103
120 void printPeriod(Period period);
121
136 void clearInputBuffer();
137
156 void fillPeriod(Period *period);
157
158 #endif // UTILS_H
```

Índice