Projeto C LI3 - Sistema de Gestão de Vendas

Generated by Doxygen 1.8.17

Chapter 1

File Index

1.1 File List

Here is a list of all files with brief descriptions:

Chapter 2

File Documentation

2.1 clients.h File Reference

Modulo que faz o tratamento dos clientes.

```
#include <stdio.h>
#include <string.h>
#include <gmodule.h>
```

Typedefs

- typedef struct clients * CLIENTS

  Estrutura dos Clientes.

Functions

- CLIENTS initClients ()

  Inicializador da Estrutura dos Clientes.
- void destroyClients (CLIENTS clients)

  Destruiçao da Estrutura dos Clientes.
- void addClient (CLIENTS clients, char *client)

  Adiciona um cliente a estrutura dos Clientes.
- void changeBranchClients (CLIENTS clients, char *client, int branch)

  Altera valor do array branch, na estrutura Cliente, para que este esteja marcado como comprou um produto num filial.
- int listOfClients (CLIENTS clients, char *client)

  Verifica se um cliente esta na Estrutura de Clientes.
- char ** findClientesOfAllBranches (CLIENTS clients, int *size)

  Função que vai buscar todos os clientes que efetuaram compras em todas as filias e ordena-os.
- int sortByLetter (const void *a, const void *b)

  Função auxiliar que ordena clientes.
- char ** initCLS (int size)

  Função que inicializa um lista de strings.
- void numberOfClientsOfAllBranches (gpointer key, gpointer value, gpointer data)

  Função auxiliar que conta o numero de clientes que efetuaram compras em todas as filias.
- int findClientsNeverBoughtCount (CLIENTS clients)

  Função que conta o numero de clientes que nunca compraram.
- void numberOfClientsNeverBought (gpointer key, gpointer value, gpointer data)

  Função auxiliar que conta o numero de clientes que nunca compraram.

2.1.1 Detailed Description

Modulo que faz o tratamento dos clientes.

2.1.2 Typedef Documentation

2.1.2.1 CLIENTS

```
typedef struct clients* CLIENTS
```

Estrutura dos Clientes.

2.1.3 Function Documentation

2.1.3.1 addClient()

```
void addClient (
            CLIENTS clients,
            char * client )
```

Adiciona um cliente a estrutura dos Clientes.

Parameters

Estrutura	a dos clientes
Cliente	a ser introduzido

2.1.3.2 changeBranchClients()

```
void changeBranchClients (
            CLIENTS clients,
            char * client,
            int branch )
```

Altera valor do array branch, na estrutura Cliente, para que este esteja marcado como comprou um produto num filial.

Parameters

Estrutura	a dos clientes
Cliente	a ser alterado
Branch	a ser alterada

2.1.3.3 destroyClients()

```
void destroyClients (
            CLIENTS clients )
```

Destruiçao da Estrutura dos Clientes.

Parameters

Estrutura	a ser destruida

2.1.3.4 findClientesOfAllBranches()

```
char** findClientesOfAllBranches (
            CLIENTS clients,
            int * size )
```

Função que vai buscar todos os clientes que efetuaram compras em todas as filias e ordena-os.

Parameters

Estrutura	dos clientes
Apontador	do tamanho do numero de clientes

2.1.3.5 findClientsNeverBoughtCount()

```
int findClientsNeverBoughtCount (
            CLIENTS clients )
```

Função que conta o numero de clientes que nunca compraram.

Parameters

Estrutura	dos Clientes

Returns

Numero de clientes que nunca compraram

2.1.3.6 initClients()

```
CLIENTS initClients ( )
```

Inicializador da Estrutura dos Clientes.

Returns

Estrutura dos Clientes

2.1.3.7 initCLS()

```
char** initCLS (
            int size )
```

Função que inicializa um lista de strings.

Parameters

Tamanho	da lista

Returns

Lista de strings inicializada

2.1.3.8 listOfClients()

```
int listOfClients (
            CLIENTS clients,
            char * client )
```

Verifica se um cliente esta na Estrutura de Clientes.

Parameters

Estrutura	a dos clientes
Cliente	a ser verificado

2.1.3.9 numberOfClientsNeverBought()

```
void numberOfClientsNeverBought (
            gpointer key,
```

```
            gpointer value,
            gpointer data )
```

Função auxiliar que conta o numero de clientes que nunca compraram.

Parameters

Key	correspondente ao cliente
Value	correspondente ao cliente
Data	intrudozida (neste caso o numero de clientes que nunca compraram)

2.1.3.10 numberOfClientsOfAllBranches()

```
void numberOfClientsOfAllBranches (
            gpointer key,
            gpointer value,
            gpointer data )
```

Função auxiliar que conta o numero de clientes que efetuaram compras em todas as filias.

Parameters

Key	correspondente ao cliente
Value	correspondente ao cliente
Data	intrudozida (neste caso o tamanho que ir tomar a lista de strings)

2.1.3.11 sortByLetter()

```
int sortByLetter (
            const void * a,
            const void * b )
```

Função auxiliar que ordena clientes.

Parameters

Cliente	1
Cliente	2

Returns

Retorna (0) caso sejam igual, (1) se cliente 2 > cliente 1 e (-1) de cliente 2 < cliente 1

2.2 files.h File Reference

Modulo de tratamento dos ficheiros.

```
#include <stdio.h>
#include <string.h>
#include "defines.h"
```

Functions

- void getFiles (char *clientsFilePath, char *productsFilePath, char *salesFilePath)

  Busca os ficheiros a serem lidos.
- int verifyFiles (char *files, char *filePath)

  Verifica se o ficheiro é valido.

2.2.1 Detailed Description

Modulo de tratamento dos ficheiros.

2.2.2 Function Documentation

2.2.2.1 getFiles()

```
void getFiles (
            char * clientsFilePath,
            char * productsFilePath,
            char * salesFilePath )
```

Busca os ficheiros a serem lidos.

Parameters

Ficheiro	dos Clientes
Ficheiro	dos Produtos
Ficheiro	das Vendas

2.2.2.2 verifyFiles()

```
int verifyFiles (
            char * files,
            char * filePath )
```

Verifica se o ficheiro é valido.

Parameters

Ficheiro	a ser testado
String	que guarda o file caso seja valido

2.3 gets.h File Reference

Modulo de tratamento de interaçao com o utilizador.

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include "sgv.h"
#include "defines.h"
#include "menu.h"
```

Functions

- char getLetter ()

  Função que pede uma letra ao utilizador.
- void getProduct (SGV sgv, char product[ ])

  Função que pede uma produto ao utilizador.
- int getMonth ()

  Função que pede um mes ao utilizador.
- int getBranch ()

  Função que pede uma filial ao utilizador.
- void getClient (SGV sgv, char client[ ])

  Função que pede uma cliente ao utilizador.
- int getMaxMonth ()

  Funcao que pede um mes ao utilizador para fazer um intervalo.
- int getOneBranch ()

  Função que pede uma filial ao utilizador.
- int getNumber ()

  Função que pede um numero ao utilizador.

2.3.1 Detailed Description

Modulo de tratamento de interaçao com o utilizador.

2.3.2 Function Documentation

2.3.2.1 getBranch()

```
int getBranch ( )
```

Função que pede uma filial ao utilizador.

Returns

Filial introduzido

2.3.2.2 getClient()

```
void getClient (
            SGV sgv,
            char client[] )
```

Função que pede uma cliente ao utilizador.

Parameters

Estrutura	de Dados para vereficar a existencia do cliente
Cliente	a ser devolvido

2.3.2.3 getLetter()

```
char getLetter ( )
```

Função que pede uma letra ao utilizador.

Returns

Letra a ser devolvida

2.3.2.4 getMaxMonth()

```
int getMaxMonth ( )
```

Funcao que pede um mes ao utilizador para fazer um intervalo.

Returns

Mes maximo

2.3.2.5 getMonth()

```
int getMonth ( )
```

Função que pede um mes ao utilizador.

Returns

Mes introduzido

2.3.2.6 getNumber()

```
int getNumber ( )
```

Função que pede um numero ao utilizador.

Returns

Numero introduzido

2.3.2.7 getOneBranch()

```
int getOneBranch ( )
```

Função que pede uma filial ao utilizador.

Returns

Filial introduzido

2.3.2.8 getProduct()

```
void getProduct (
            SGV sgv,
            char product[] )
```

Função que pede uma produto ao utilizador.

Parameters

Estrutura	de Dados para vereficar a existencia do produto
Produto	a ser devolvido

2.4 interface.h File Reference

Modulo que faz o tratamento das queries.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "sgv.h"
#include "defines.h"
#include "menu.h"
#include "files.h"
#include "gets.h"
#include "sales.h"
#include "products.h"
```

Functions

- SGV initSGV ()

  Inicialização da Base de Dados.
- void destroySGV (SGV sgv)

  Destrução da Base de Dados.
- SGV loadSGVFromFiles (SGV sgv, char ∗clientsFilePath, char ∗productsFilePath, char ∗salesFilePath)

  Carregar Base de Dados (SGV) a partir de ficheiros.
- void getCurrentFilesInfo (SGV sgv)
- void getProductsStartedByLetter (SGV sgv, char letter)

  Função que vai buscar todos produtos que começam por um letra (QUERY2)
- void getProductSalesAndProfit (SGV sgv, char ∗productID, int month)

  Função que vai buscar todas as vendas e faturaçao ocurridas num certo mes por um produto(QUERY3)
- void getProductsNeverBought (SGV sgv, int branchID)

  Função que vai buscar todos os produtos que nao foram vendidos (por filiais ou geral) (QUERY4)
- void getClientsOfAllBranches (SGV sgv)

  Função que vai buscar todos os clientes que efetuaram compras em todas as filias (QUERY5)
- void getClientsAndProductsNeverBoughtCount (SGV sgv)

  Função que vai calcular o numero de produtos e clientes que nao efetuaram compras (QUERY6)
- void getProductsBoughtByClient (SGV sgv, char ∗clientID)

  Função que vai buscar o numero de compras de um determinado cliente (QUERY7)
- void getSalesAndProfit (SGV sgv, int minMonth, int maxMonth)

  Função que vai buscar o numero de compras e a faturação num intervalo de meses (QUERY8)
- void getProductsBuyers (SGV sgv, char ∗productID, int branch)

  Função que vai buscar os clientes que compraram um produto num certo filial (QUERY9)
- void getClientFavoriteProducts (SGV sgv, char ∗clientID, int month)

  Função que vai buscar os produtos comprados por um cliente num certo mes, ordenando-os (QUERY10)
- void getTopSelledProducts (SGV sgv, int limit)

  Função que vai buscar os produtos mais vendidos (QUERY11)
- void getClientTopProfitProducts (SGV sgv, char ∗clientID, int limit)

  Função que encontra os 'n' produtos mais comprados pelo cliente (QUERY12)
- void showFilesInfo (SGV sgv, char ∗clientsFilePath, char ∗productsFilePath, char ∗salesFilePath, float time)

  Função que mostra a informaçao sobre a leitura dos files da primeira query (QUERY13)

2.4.1 Detailed Description

Modulo que faz o tratamento das queries.

2.4.2 Function Documentation

2.4.2.1 destroySGV()

```
void destroySGV (
            SGV sgv )
```

Destrução da Base de Dados.

Parameters

Base	de Dados a ser destruida

2.4.2.2 getClientFavoriteProducts()

```
void getClientFavoriteProducts (
            SGV sgv,
            char * clientID,
            int month )
```

Função que vai buscar os produtos comprados por um cliente num certo mes, ordenando-os (QUERY10)

Parameters

Estrutura	de Dados (SGV)
Cliente	a ser procurado
Mes	a ser procurada

2.4.2.3 getClientsAndProductsNeverBoughtCount()

```
void getClientsAndProductsNeverBoughtCount (
            SGV sgv )
```

Função que vai calcular o numero de produtos e clientes que nao efetuaram compras (QUERY6)

Parameters

Estrutura	de Dados (SGV)

2.4.2.4 getClientsOfAllBranches()

```
void getClientsOfAllBranches (
            SGV sgv )
```

Função que vai buscar todos os clientes que efetuaram compras em todas as filias (QUERY5)

Parameters

Estrutura	de Dados (SGV)

2.4.2.5 getClientTopProfitProducts()

```
void getClientTopProfitProducts (
            SGV sgv,
            char * clientID,
            int limit )
```

Função que encontra os 'n' produtos mais comprados pelo cliente (QUERY12)

Parameters

Estrutura	de Dados (SGV)
Cliente	em questao
Numero	de produtos a pesquisar

2.4.2.6 getCurrentFilesInfo()

```
void getCurrentFilesInfo (
            SGV sgv )
```

2.4.2.7 getProductSalesAndProfit()

```
void getProductSalesAndProfit (
            SGV sgv,
```

```
char * productID,
int month )
```

Função que vai buscar todas as vendas e faturaçao ocurridas num certo mes por um produto(QUERY3)

Parameters

Estrutura	de Dados (SGV)
Produto	a ser examinado
Mes	em questao

2.4.2.8 getProductsBoughtByClient()

```
void getProductsBoughtByClient (
            SGV sgv,
            char * clientID )
```

Função que vai buscar o numero de compras de um determinado cliente (QUERY7)

Parameters

Estrutura	de Dados (SGV)
Cliente	a ser examinado

2.4.2.9 getProductsBuyers()

```
void getProductsBuyers (
            SGV sgv,
            char * productID,
            int branch )
```

Função que vai buscar os clientes que compraram um produto num certo filial (QUERY9)

Parameters

Estrutura	de Dados (SGV)
Produto	a ser procurado
Filial	a ser procurada

2.4.2.10 getProductsNeverBought()

```
void getProductsNeverBought (
            SGV sgv,
            int branchID )
```

Função que vai buscar todos os produtos que nao foram vendidos (por filiais ou geral) (QUERY4)

Parameters

Estrutura	de Dados (SGV)
Filial	a procurar produtos

2.4.2.11 getProductsStartedByLetter()

```
void getProductsStartedByLetter (
            SGV sgv,
            char letter )
```

Função que vai buscar todos produtos que começam por um letra (QUERY2)

Parameters

Estrutura	de Dados (SGV)
Letra	inicial do produto

2.4.2.12 getSalesAndProfit()

```
void getSalesAndProfit (
            SGV sgv,
            int minMonth,
            int maxMonth )
```

Função que vai buscar o numero de compras e a faturação num intervalo de meses (QUERY8)

Parameters

Estrutura	de Dados (SGV)
Mes	minimo
Mes	maximo

2.4.2.13 getTopSelledProducts()

```
void getTopSelledProducts (
            SGV sgv,
            int limit )
```

Função que vai buscar os produtos mais vendidos (QUERY11)

Parameters

Estrutura	de Dados (SGV)
Numero	de produtos a pesquisar

2.4.2.14 initSGV()

```
SGV initSGV ( )
```

Inicialização da Base de Dados.

Returns

Base de Dados

2.4.2.15 loadSGVFromFiles()

```
SGV loadSGVFromFiles (
            SGV sgv,
            char * clientsFilePath,
            char * productsFilePath,
            char * salesFilePath )
```

Carregar Base de Dados (SGV) a partir de ficheiros.

Parameters

Ficheiro	dos Clientes
Ficheiro	dos Produtos
Ficheiro	das Vendas

Returns

Base de Dados (SGV) carregada

2.4.2.16 showFilesInfo()

```
void showFilesInfo (
            SGV sgv,
            char * clientsFilePath,
            char * productsFilePath,
            char * salesFilePath,
            float time )
```

Função que mostra a informaçao sobre a leitura dos files da primeira query (QUERY13)

Parameters

Estrutura	de Dados (SGV)
Ficheiro	dos clientes
Ficheiro	dos produtos
Ficheiro	das vendas
Tempo	de construção da base de dados

2.5 menu.h File Reference

Modulo de tratamento do menu.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include "defines.h"
#include "sgv.h"
#include "products.h"
#include "clients.h"
```

Functions

- int showMenuOption ()
- int showMenu1 (int valid)
- int showAndGetOptionMenu2 (int valid)
- void showMenu2 (int op)
- void parteDeCimaMenu ()
- void parteDeBaixoMenu ()
- void letrasSGV ()
- void menuPalavra ()
- void loading ()
- void parteDeCimaTabela ()
- void parteDeBaixoTabela ()
- void infoPalavra ()
- void dataPalavra ()
- void parteDeCimaTempos ()
- void parteDeBaixoTempos ()
- void getEnter ()
- void enterToContinue ()
- void mostraLetra (char letter)
- void mostraProdutosQuery2 (char ∗∗prods, int size, int nPags, int pag)
- void miniMenuQuery2 (float time, int nPags, int pag, int size)
- void produtoMesPalavras (char ∗product, int month)
- void filiaisPalavras ()
- void fatUniPalavras ()
- void dadosCompras (double res[ ][4])
- void visaoGlobalPalvra ()
- void FatUniPalavrasG ()
- void dadosComprasG (double res[ ][4])

- void miniMenuQuery3 (float time, int op)
- void mostraFilial (int filial)
- void mostraClientesQuery5 (char **clients, int size, int nPags, int pag)
- void setasPalavras ()
- void miniMenuQuery5 (float time, int size, int nPags, int pag)
- void mostraContador (int res[ ])
- void miniMenuQuery6 (float time)
- void clientePalavra (char *client)
- void dadosClientesMes (int res[ ][12])
- void dadosTotalVendas (int *intervalo, double *res)
- void printaDadosTotalVendas (double res, int q)
- void produtoFilialPalavras (char *produto, int filial)
- void clientesQuery9 (char **clientesN, int unidN, char **clientesP, int unidP, int maior)
- void miniMenuQuery9 (int total, float time)
- void clienteMesPalavras (char *client, int month)
- void dadosQuery10 (char **prods, int qt[ ], int size)
- void query11Palavras ()
- void dadosQuery11 (char **prods, int data[ ][3][2], int size, int nPags, int pag)
- void mostraClientesPalavras (char *client)
- void dadosQuery12 (char **prods, double qt[ ], int limit, int pag)
- void showProductsStartbyLetter (char **prods, char letter, int size, float time)
- void showProductSalesAndProfit (char *product, int month, double res[ ][4], float time)
- void showProductsNeverBought (char **prods, int size, int branchID, float time)
- void showClientsOfAllBranches (char **clients, int size, float time)
- void showClientsAndProductsNeverBoughtCount (int res[ ], float time)
- void showProductsBoughtByClient (char *client, int res[ ][12], float time)
- void showSalesAndProfit (int minMonth, int maxMonth, double res[ ], float time)
- void showProductsBuyers (char *productID, int branch, char **clientsN, char **clientsP, int unitsN, int unitsP, float time)
- void showClientFavoriteProducts (char **prods, int qt[ ], char *client, int month, int size, float time)
- void showTopSoldProducts (char **prods, int data[ ][3][2], int limit, float time)
- void showClientTopProfitProducts (char **prods, double qt[ ], char *client, int limit, float time)
- void filesInfo (int l[ ], char *clients, char *prods, char *sales, float time)
- void showDestructionTime (float time)
- char getLetter2 ()
- void getProductAndMonth2 (SGV sgv, char product[ ], int *month)
- int getBranches2 ()
- void getClient2 (SGV sgv, char client[ ])
- void getMinAndMaxMonth2 (int *minMonth, int *maxMonth)
- void getProductAndBranch2 (SGV sgv, char *product, int *branch)
- void getClientAndMonth2 (SGV sgv, char *client, int *month)
- int getNumber2 ()
- int getClientAndNumber (SGV sgv, char *client)

2.5.1 Detailed Description

Modulo de tratamento do menu.

2.5.2 Function Documentation

2.5.2.1 clienteMesPalavras()

```
void clienteMesPalavras (
            char * client,
            int month )
```

2.5.2.2 clientePalavra()

```
void clientePalavra (
            char * client )
```

2.5.2.3 clientesQuery9()

```
void clientesQuery9 (
            char ** clientesN,
            int unidN,
            char ** clientesP,
            int unidP,
            int maior )
```

2.5.2.4 dadosClientesMes()

```
void dadosClientesMes (
            int res[][12] )
```

2.5.2.5 dadosCompras()

```
void dadosCompras (
            double res[][4] )
```

2.5.2.6 dadosComprasG()

```
void dadosComprasG (
            double res[][4] )
```

2.5.2.7 dadosQuery10()

```
void dadosQuery10 (
            char ** prods,
            int qt[],
            int size )
```

2.5.2.8 dadosQuery11()

```
void dadosQuery11 (
            char ** prods,
            int data[][3][2],
            int size,
            int nPags,
            int pag )
```

2.5.2.9 dadosQuery12()

```
void dadosQuery12 (
            char ** prods,
            double qt[],
            int limit,
            int pag )
```

2.5.2.10 dadosTotalVendas()

```
void dadosTotalVendas (
            int * intervalo,
            double * res )
```

2.5.2.11 dataPalavra()

```
void dataPalavra ( )
```

2.5.2.12 enterToContinue()

```
void enterToContinue ( )
```

2.5.2.13 fatUniPalavras()

```
void fatUniPalavras ( )
```

2.5.2.14 FatUniPalavrasG()

```
void FatUniPalavrasG ( )
```

2.5.2.15 filesInfo()

```
void filesInfo (
             int l[],
             char * clients,
             char * prods,
             char * sales,
             float time )
```

2.5.2.16 filiaisPalavras()

```
void filiaisPalavras ( )
```

2.5.2.17 getBranches2()

```
int getBranches2 ( )
```

2.5.2.18 getClient2()

```
void getClient2 (
             SGV sgv,
             char client[] )
```

2.5.2.19 getClientAndMonth2()

```
void getClientAndMonth2 (
             SGV sgv,
             char * client,
             int * month )
```

2.5.2.20 getClientAndNumber()

```
int getClientAndNumber (
            SGV sgv,
            char * client )
```

2.5.2.21 getEnter()

```
void getEnter ( )
```

2.5.2.22 getLetter2()

```
char getLetter2 ( )
```

2.5.2.23 getMinAndMaxMonth2()

```
void getMinAndMaxMonth2 (
            int * minMonth,
            int * maxMonth )
```

2.5.2.24 getNumber2()

```
int getNumber2 ( )
```

2.5.2.25 getProductAndBranch2()

```
void getProductAndBranch2 (
            SGV sgv,
            char * product,
            int * branch )
```

2.5.2.26 getProductAndMonth2()

```
void getProductAndMonth2 (
            SGV sgv,
            char product[],
            int * month )
```

2.5.2.27 infoPalavra()

```
void infoPalavra ( )
```

2.5.2.28 letrasSGV()

```
void letrasSGV ( )
```

2.5.2.29 loading()

```
void loading ( )
```

2.5.2.30 menuPalavra()

```
void menuPalavra ( )
```

2.5.2.31 miniMenuQuery2()

```
void miniMenuQuery2 (
            float time,
            int nPags,
            int pag,
            int size )
```

2.5.2.32 miniMenuQuery3()

```
void miniMenuQuery3 (
            float time,
            int op )
```

2.5.2.33 miniMenuQuery5()

```
void miniMenuQuery5 (
            float time,
            int size,
            int nPags,
            int pag )
```

2.5.2.34 miniMenuQuery6()

```
void miniMenuQuery6 (
            float time )
```

2.5.2.35 miniMenuQuery9()

```
void miniMenuQuery9 (
            int total,
            float time )
```

2.5.2.36 mostraClientesPalavras()

```
void mostraClientesPalavras (
            char * client )
```

2.5.2.37 mostraClientesQuery5()

```
void mostraClientesQuery5 (
            char ** clients,
            int size,
            int nPags,
            int pag )
```

2.5.2.38 mostraContador()

```
void mostraContador (
            int res[] )
```

2.5.2.39 mostraFilial()

```
void mostraFilial (
            int filial )
```

2.5.2.40 mostraLetra()

```
void mostraLetra (
            char letter )
```

2.5.2.41 mostraProdutosQuery2()

```
void mostraProdutosQuery2 (
            char ** prods,
            int size,
            int nPags,
            int pag )
```

2.5.2.42 parteDeBaixoMenu()

```
void parteDeBaixoMenu ( )
```

2.5.2.43 parteDeBaixoTabela()

```
void parteDeBaixoTabela ( )
```

2.5.2.44 parteDeBaixoTempos()

```
void parteDeBaixoTempos ( )
```

2.5.2.45 parteDeCimaMenu()

```
void parteDeCimaMenu ( )
```

2.5.2.46 parteDeCimaTabela()

```
void parteDeCimaTabela ( )
```

2.5.2.47 parteDeCimaTempos()

```
void parteDeCimaTempos ( )
```

2.5.2.48 printaDadosTotalVendas()

```
void printaDadosTotalVendas (
            double res,
            int q )
```

2.5.2.49 produtoFilialPalavras()

```
void produtoFilialPalavras (
            char * produto,
            int filial )
```

2.5.2.50 produtoMesPalavras()

```
void produtoMesPalavras (
            char * product,
            int month )
```

2.5.2.51 query11Palavras()

```
void query11Palavras ( )
```

2.5.2.52 setasPalavras()

```
void setasPalavras ( )
```

2.5.2.53 showAndGetOptionMenu2()

```
int showAndGetOptionMenu2 (
            int valid )
```

2.5.2.54 showClientFavoriteProducts()

```
void showClientFavoriteProducts (
            char ** prods,
            int qt[],
            char * client,
            int month,
            int size,
            float time )
```

2.5.2.55 showClientsAndProductsNeverBoughtCount()

```
void showClientsAndProductsNeverBoughtCount (
            int res[],
            float time )
```

2.5.2.56 showClientsOfAllBranches()

```
void showClientsOfAllBranches (
            char ** clients,
            int size,
            float time )
```

2.5.2.57 showClientTopProfitProducts()

```
void showClientTopProfitProducts (
            char ** prods,
            double qt[],
            char * client,
            int limit,
            float time )
```

2.5.2.58 showDestructionTime()

```
void showDestructionTime (
            float time )
```

2.5.2.59 showMenu1()

```
int showMenu1 (
            int valid )
```

2.5.2.60 showMenu2()

```
void showMenu2 (
            int op )
```

2.5.2.61 showMenuOption()

```
int showMenuOption ( )
```

2.5.2.62 showProductSalesAndProfit()

```
void showProductSalesAndProfit (
            char * product,
            int month,
            double res[][4],
            float time )
```

2.5.2.63 showProductsBoughtByClient()

```
void showProductsBoughtByClient (
            char * client,
            int res[][12],
            float time )
```

2.5.2.64 showProductsBuyers()

```
void showProductsBuyers (
            char * productID,
            int branch,
            char ** clientsN,
            char ** clientsP,
            int unitsN,
            int unitsP,
            float time )
```

2.5.2.65 showProductsNeverBought()

```
void showProductsNeverBought (
            char ** prods,
            int size,
            int branchID,
            float time )
```

2.5.2.66 showProductsStartbyLetter()

```
void showProductsStartbyLetter (
            char ** prods,
            char letter,
            int size,
            float time )
```

2.5.2.67 showSalesAndProfit()

```
void showSalesAndProfit (
            int minMonth,
            int maxMonth,
            double res[],
            float time )
```

2.5.2.68 showTopSoldProducts()

```
void showTopSoldProducts (
            char ** prods,
            int data[][3][2],
            int limit,
            float time )
```

2.5.2.69 visaoGlobalPalvra()

```
void visaoGlobalPalvra ( )
```

2.6 products.h File Reference

Modulo que faz o tratamento dos produtos.

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

Typedefs

- typedef struct products * PRODUCTS

  Estrutura dos Produtos.
- typedef struct productSearch * PRODUCTSEARCH

  Estrutura que guarda os produtos durante uma procura.

Functions

- PRODUCTS initProducts ()

  Inicializador da Estrutura dos Produtos.
- void destroyProducts (PRODUCTS products)

  Destruiçao da Estrutura dos Produtos.
- void addProduct (PRODUCTS products, char *product)

  Adiciona um produto a estrutura dos Produtos.
- void changeBranchProducts (PRODUCTS products, char *product, int branch)

  Altera value do produto, indicando em qual filial o produto foi vendido.
- int listOfProducts (PRODUCTS products, char *product)

  Verifica se um produto esta na Estrutura de Produtos.
- PRODUCTSEARCH findProductsByLetter (PRODUCTS products)

  Função que encontra os produtos começados por uma determinada letra.
- int iter_all (gpointer key, gpointer value, gpointer data)

  Função auxiliar que atravesa todos os nodos de uma arvore.
- PRODUCTSEARCH initProductSearch (int size)

  Inicializa uma estrutura de procura para os produtos.
- char ** getProductsOfPS (PRODUCTSEARCH ps)

  Função que passa os produtos encontrados pela estrutura PRODUCTSEARCH para um lista de strings.
- int getTotalOfPS (PRODUCTSEARCH ps)

  Função que retorna o numero de produtos encontrados pela estrutura PRODUCTSEARCH.
- void destroyPS (PRODUCTSEARCH ps)

  Função que destroi a estrutura PRODUCTSEARCH.
- PRODUCTSEARCH findProductsNeverBought (PRODUCTS *products, int branch)

  Função que encontra todos os produtos nao vendidos de uma ou todas filial.
- int productsNeverBought0 (gpointer key, gpointer value, gpointer data)

  Função auxiliar que atravesa todos os nodos de uma arvore e encontra os produtos nao vendidos na filial 1.
- int productsNeverBought1 (gpointer key, gpointer value, gpointer data)

  Função auxiliar que atravesa todos os nodos de uma arvore e encontra os produtos nao vendidos na filial 2.
- int productsNeverBought2 (gpointer key, gpointer value, gpointer data)

  Função auxiliar que atravesa todos os nodos de uma arvore e encontra os produtos nao vendidos ma filial 3.
- int sizeOfProductsNeverBoughtEveryBranch (gpointer key, gpointer value, gpointer data)

  Função auxiliar que atravesa todos os nodos de uma arvore e calcula o numero de produtos nao vendidos em todas a filiais.
- int sizeOfProductsNeverBoughtAllBranches (gpointer key, gpointer value, gpointer data)

  Função auxiliar que atravesa todos os nodos de uma arvore e calcula o numero de produtos nao vendidos.
- int productsNeverBought (gpointer key, gpointer value, gpointer data)

  Função auxiliar que atravesa todos os nodos de uma arvore e encontra os produtos nao vendidos em todas as filiais.
- int findProductsNeverBoughtCount (PRODUCTS *products)

  Função que conta o numero de produtos que nunca foram comprados.
- int numberOfProductsNeverBought (gpointer key, gpointer value, gpointer data)

  Função auxiliar que conta o numero de produtos que nunca foram comprados.

2.6.1 Detailed Description

Modulo que faz o tratamento dos produtos.

2.6.2 Typedef Documentation

2.6.2.1 PRODUCTS

```
typedef struct products* PRODUCTS
```

Estrutura dos Produtos.

2.6.2.2 PRODUCTSEARCH

```
typedef struct productSearch* PRODUCTSEARCH
```

Estrutura que guarda os produtos durante uma procura.

2.6.3 Function Documentation

2.6.3.1 addProduct()

```
void addProduct (
            PRODUCTS products,
            char * product )
```

Adiciona um produto a estrutura dos Produtos.

Parameters

Estrutura	a dos produtos
Produto	a ser introduzido

2.6.3.2 changeBranchProducts()

```
void changeBranchProducts (
            PRODUCTS products,
```

```
            char * product,
            int branch )
```

Altera value do produto, indicando em qual filial o produto foi vendido.

Parameters

Estrutura	a dos produtos
Produto	a ser aleterado
Filial	em que a venda foi efetuada

2.6.3.3 destroyProducts()

```
void destroyProducts (
            PRODUCTS products )
```

Destruiçao da Estrutura dos Produtos.

Parameters

Estrutura	a ser destruida

2.6.3.4 destroyPS()

```
void destroyPS (
            PRODUCTSEARCH ps )
```

Função que destroi a estrutura PRODUCTSEARCH.

Parameters

Estrutura	PRODUCTSEARCH a ser eleminada

2.6.3.5 findProductsByLetter()

```
PRODUCTSEARCH findProductsByLetter (
            PRODUCTS products )
```

Função que encontra os produtos começados por uma determinada letra.

Parameters

Estrutura	a dos produtos

Returns

Estrutura a qual guarda os produtos procurados

2.6.3.6 findProductsNeverBought()

```
PRODUCTSEARCH findProductsNeverBought (
            PRODUCTS * products,
            int branch )
```

Função que encontra todos os produtos nao vendidos de uma ou todas filial.

Parameters

Array	de Estrutura de Produtos
Filial	a procurar

Returns

Array de Estruturas PRODUCTSEARCH

2.6.3.7 findProductsNeverBoughtCount()

```
int findProductsNeverBoughtCount (
            PRODUCTS * products )
```

Função que conta o numero de produtos que nunca foram comprados.

Parameters

Array	de Estrutura dos Produtos

Returns

Numero de produtos que nunca foram comprados

2.6.3.8 getProductsOfPS()

```
char** getProductsOfPS (
            PRODUCTSEARCH ps )
```

Função que passa os produtos encontrados pela estrutura PRODUCTSEARCH para um lista de strings.

Parameters

Estrutura	PRODUCTSEARCH

Returns

Lista de Strings

2.6.3.9 getTotalOfPS()

```
int getTotalOfPS (
            PRODUCTSEARCH ps )
```

Função que retorna o numero de produtos encontrados pela estrutura PRODUCTSEARCH.

Parameters

Estrutura	PRODUCTSEARCH

Returns

Numero de produtos lidos

2.6.3.10 initProducts()

```
PRODUCTS initProducts ( )
```

Inicializador da Estrutura dos Produtos.

Returns

Estrutura dos Produtos

2.6.3.11 initProductSearch()

```
PRODUCTSEARCH initProductSearch (
            int size )
```

Inicializa uma estrutura de procura para os produtos.

Parameters

Tamanho	da lista a strings

Returns

Estrutura inicializada

2.6.3.12 iter_all()

```
int iter_all (
            gpointer key,
            gpointer value,
            gpointer data )
```

Função auxiliar que atravesa todos os nodos de uma arvore.

Parameters

Key	correspondente ao nodo
Value	correspondente ao nodo
Paramentro	passado pela funçao "g_hash_table_foreach" (neste caso a estrutura PRODUCTSEARCH)

Returns

Inteiro caso queria parar a funçao

2.6.3.13 listOfProducts()

```
int listOfProducts (
            PRODUCTS products,
            char * product )
```

Verifica se um produto esta na Estrutura de Produtos.

Parameters

Estrutura	a dos produtos
Produto	a ser verificado

2.6.3.14 numberOfProductsNeverBought()

```
int numberOfProductsNeverBought (
            gpointer key,
            gpointer value,
            gpointer data )
```

Função auxiliar que conta o numero de produtos que nunca foram comprados.

Parameters

Key	correspondente ao produto
Value	correspondente ao produto
Data	intrudozida (neste caso o numero de produtos que nunca foram comprados)

Returns

Inteiro caso queria parar a funçao

2.6.3.15 productsNeverBought()

```
int productsNeverBought (
            gpointer key,
            gpointer value,
            gpointer data )
```

Função auxiliar que atravesa todos os nodos de uma arvore e encontra os produtos nao vendidos em todas as filiais.

Parameters

Key	correspondente ao nodo
Value	correspondente ao nodo
Paramentro	passado pela funçao "g_hash_table_foreach" (neste caso um array de inteiros)

Returns

Inteiro caso queria parar a funçao

2.6.3.16 productsNeverBought0()

```
int productsNeverBought0 (
            gpointer key,
            gpointer value,
            gpointer data )
```

Função auxiliar que atravesa todos os nodos de uma arvore e encontra os produtos nao vendidos na filial 1.

Parameters

Key	correspondente ao nodo
Value	correspondente ao nodo
Paramentro	passado pela funçao "g_hash_table_foreach" (neste caso um array de inteiros)

Returns

Inteiro caso queria parar a funçao

2.6.3.17 productsNeverBought1()

```
int productsNeverBought1 (
            gpointer key,
            gpointer value,
            gpointer data )
```

Função auxiliar que atravesa todos os nodos de uma arvore e encontra os produtos nao vendidos na filial 2.

Parameters

Key	correspondente ao nodo
Value	correspondente ao nodo
Paramentro	passado pela funçao "g_hash_table_foreach" (neste caso um array de inteiros)

Returns

Inteiro caso queria parar a funçao

2.6.3.18 productsNeverBought2()

```
int productsNeverBought2 (
            gpointer key,
            gpointer value,
            gpointer data )
```

Função auxiliar que atravesa todos os nodos de uma arvore e encontra os produtos nao vendidos ma filial 3.

Parameters

Key	correspondente ao nodo
Value	correspondente ao nodo
Paramentro	passado pela funçao "g_hash_table_foreach" (neste caso um array de inteiros)

Returns

Inteiro caso queria parar a funçao

2.6.3.19 sizeOfProductsNeverBoughtAllBranches()

```
int sizeOfProductsNeverBoughtAllBranches (
            gpointer key,
            gpointer value,
            gpointer data )
```

Função auxiliar que atravesa todos os nodos de uma arvore e calcula o numero de produtos nao vendidos.

Parameters

Key	correspondente ao nodo
Value	correspondente ao nodo
Paramentro	passado pela funçao "g_hash_table_foreach" (neste caso um array de inteiros)

Returns

Inteiro caso queria parar a funçao

2.6.3.20 sizeOfProductsNeverBoughtEveryBranch()

```
int sizeOfProductsNeverBoughtEveryBranch (
            gpointer key,
            gpointer value,
            gpointer data )
```

Função auxiliar que atravesa todos os nodos de uma arvore e calcula o numero de produtos nao vendidos em todas a filiais.

Parameters

Key	correspondente ao nodo
Value	correspondente ao nodo
Paramentro	passado pela funçao "g_hash_table_foreach" (neste caso um array de inteiros)

Returns

Inteiro caso queria parar a funçao

2.7 sales.h File Reference

Modulo que faz o tratamento das vendas.

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

Typedefs

- typedef struct sales * SALES

  Estrutura das vendas.
- typedef struct prodCl * PRODCL

  Estrutura que relaciona produtos com clientes.
- typedef struct clProd * CLPROD

  Estrutura que relaciona clientes com produtos.
- typedef struct clSearcher * CLSEARCHER

  Estrutura que guarda clientes numa procura.
- typedef struct prSearcher * PRSEARCHER

  Estrutura que guarda produtos numa procura.

Functions

- SALES initSales ()

  Inicializador da Estrutura das Vendas.
- void destroySales (SALES sales)

  Destruiçao da Estrutura das Vendas.
- void destroyProductsClients (void *prCl)

  Destruiçao da Estrutura que relaciona produtos com clientes.
- void destroyClientsProducts (void *clPr)

  Destruiçao da Estrutura que relaciona clientes com produtos.
- void addBranch (SALES sales, char *product, char *client, char type, float price, int units, int month)

  Carrega uma venda a Estrutura de Vendas.
- CLPROD initClientProducts ()

  Inicialização da Estrutura que relaciona Clientes com Produtos.
- void changeClientsProducts (CLPROD cp, char *product, int month, int units, float price)

  Altera a Estrutura que relaciona clientes com produtos, alterando a faturaçao do produto.
- PRODCL initProductsClients ()

  Inicialização da Estrutura que relaciona Produtos com Clientes.
- void changeTotalProducts (double total[ ][2][12], char type, float price, int month)

  Altera a Estrutura que relaciona produtos com clientes, aumentando o preço e o numero de vendas sofridas por um produto.
- void changeProductsClients (PRODCL cl, char *client, char type)

  Altera a Estrutura que relaciona produtos com clientes, alterando o tipo de venda que o cliente fez.
- void findProductSalesAndProfit (SALES *sales, char *productID, int month, double res[ ][4])

  Função que encontra as vendas e a faturaçao de um produto num certo mes.
- void findProductSellesByClient (SALES sales, char *client, int *res)

  Função que encontra as vendas feitas por um cliente.
- CLSEARCHER getCLSearcher (SALES sale, char *productID)

  Função que encontra as clientes que compraram um produto.
- int numberOfSales (PRODCL pc, int flag)

  Função que encontra o numero de vendas feitas.
- void findCLS (gpointer key, gpointer value, gpointer data)

  Função auxiliar que encontra clientes.

- int getClientsAndUnitsCLS (CLSEARCHER cls, char ∗∗∗clientes, int flag)

  Função que retira os clientes e as unidades da Estrutura de procura "CLSEARCHER".
- void destroyCLS (CLSEARCHER cls)

  Função que elimina a estrutura de procura "CLSEARCHER".
- PRSEARCHER findClientFavotiteProducts (SALES ∗sales, char ∗client, int month, int ∗size)

  Função que constroi a estrutura de procura "PRSEARCHER".
- int sizeOfProducts (SALES sales, char ∗client, int month)

  Função que calcula o numero de produtos comprados por um cliente.
- int sort (const void ∗one, const void ∗two)

  Função auxilar que retorna se o segundo parametro é maior do que o primeiro.
- char ∗∗ getProductsAndUnitsPRS (PRSEARCHER prs, int size, int qt[ ])

  Função encontra os produtos da estrutura "PRSEARCHER" e coloca-os numa lista de strings.
- GHashTable ∗ mergeBranches (SALES ∗sales)

  Função que junta as filiais numa unica hash table.
- char ∗∗ findSalesMerge (GHashTable ∗merge, double qt[ ], int size, int limit, int flag)

  Função encontra os 'n' produtos que mais faturou em todo ano (este executa a query 11 e query a partir de uma flag)
- void findData (SALES ∗sales, int data[ ][3][2], char ∗∗prods, int limit, int branch)
- GHashTable ∗ mergeBranchesAndMonths (SALES ∗sales, char ∗client)

  Função que junta as filiais devididas em meses numa unica hash table.

2.7.1 Detailed Description

Modulo que faz o tratamento das vendas.

2.7.2 Typedef Documentation

2.7.2.1 CLPROD

```
typedef struct clProd* CLPROD
```

Estrutura que relaciona clientes com produtos.

2.7.2.2 CLSEARCHER

```
typedef struct clSearcher* CLSEARCHER
```

Estrutura que guarda clientes numa procura.

2.7.2.3 PRODCL

```
typedef struct prodCl* PRODCL
```

Estrutura que relaciona produtos com clientes.

2.7.2.4 PRSEARCHER

```
typedef struct prSearcher* PRSEARCHER
```

Estrutura que guarda produtos numa procura.

2.7.2.5 SALES

```
typedef struct sales* SALES
```

Estrutura das vendas.

2.7.3 Function Documentation

2.7.3.1 addBranch()

```
void addBranch (
            SALES sales,
            char * product,
            char * client,
            char type,
            float price,
            int units,
            int month )
```

Carrega uma venda a Estrutura de Vendas.

Parameters

Estrutura	de Vendas
Cliente	a adicionar
Produto	a adicionar
Tipo	da venda
Custo	unitario do produto
Unidades	vendidas
Mes	da venda

2.7.3.2 changeClientsProducts()

```
void changeClientsProducts (
            CLPROD cp,
            char * product,
            int month,
            int units,
            float price )
```

Altera a Estrutura que relaciona clientes com produtos, alterando a faturaçao do produto.

Parameters

Estrutra	a ser alterada
Produto	que sera alterado
Mes	da venda
Unidades	da vena
Preço	unitario da venda

2.7.3.3 changeProductsClients()

```
void changeProductsClients (
            PRODCL cl,
            char * client,
            char type )
```

Altera a Estrutura que relaciona produtos com clientes, alterando o tipo de venda que o cliente fez.

Parameters

Estrutra	a ser alterada
Cliente	que sera alterado
Tipo	de venda

2.7.3.4 changeTotalProducts()

```
void changeTotalProducts (
            double total[][2][12],
            char type,
            float price,
            int month )
```

Altera a Estrutura que relaciona produtos com clientes, aumentando o preço e o numero de vendas sofridas por um produto.

Parameters

Estrutra	a ser alterada
Tipo	da venda
Preço	unitario da venda
Mes	da venda

2.7.3.5 destroyClientsProducts()

```
void destroyClientsProducts (
            void * clPr )
```

Destruiçao da Estrutura que relaciona clientes com produtos.

Parameters

Estrutura	a ser destruida

2.7.3.6 destroyCLS()

```
void destroyCLS (
            CLSEARCHER cls )
```

Função que elimina a estrutura de procura "CLSEARCHER".

Parameters

Estrutura	a ser destruida

2.7.3.7 destroyProductsClients()

```
void destroyProductsClients (
            void * prCl )
```

Destruiçao da Estrutura que relaciona produtos com clientes.

Parameters

Estrutura	a ser destruida

2.7.3.8 destroySales()

```
void destroySales (
            SALES sales )
```

Destruiçao da Estrutura das Vendas.

Parameters

Estrutura	a ser destruida

2.7.3.9 findClientFavotiteProducts()

```
PRSEARCHER findClientFavotiteProducts (
            SALES * sales,
            char * client,
            int month,
            int * size )
```

Função que constroi a estrutura de procura "PRSEARCHER".

Parameters

Array	de Estrutura das vendas
Cliente	a ser procurado
Mes	a ser procurado
Numero	de produtos comprados por um cliente

Returns

Estrutura de procura "PRSEARCHER"

2.7.3.10 findCLS()

```
void findCLS (
            gpointer key,
            gpointer value,
            gpointer data )
```

Função auxiliar que encontra clientes.

Parameters

Estrutura	que relaciona produtos com clientes
Flag	para procurar clientes que fizeram compras de tipo N ou P

Returns

Numero de vendas

2.7.3.11 findData()

```
void findData (
            SALES * sales,
            int data[][3][2],
            char ** prods,
            int limit,
            int branch )
```

Parameters

Array	de estrutura das vendas
Matriz	que ir guardar os resultados dividido em filiais, vendas, e numero de clientes comprados
Numero	de produtos
Filial	a procurar

2.7.3.12 findProductSalesAndProfit()

```
void findProductSalesAndProfit (
            SALES * sales,
            char * productID,
            int month,
            double res[][4] )
```

Função que encontra as vendas e a faturaçao de um produto num certo mes.

Parameters

Estrutura	de filiais
Produto	a procura
Mes	em questao
Array	o qual sera guardado as respostas das vendas e faturaçao dividido em filiais

2.7.3.13 findProductSellesByClient()

```
void findProductSellesByClient (
            SALES sales,
            char * client,
            int * res )
```

Função que encontra as vendas feitas por um cliente.

Parameters

Estrutura	das vendas
Cliente	a procurar
Array	onde sera guardado as respostas

2.7.3.14 findSalesMerge()

```
char** findSalesMerge (
            GHashTable * merge,
            double qt[],
            int size,
            int limit,
            int flag )
```

Função encontra os 'n' produtos que mais faturou em todo ano (este executa a query 11 e query a partir de uma flag)

Parameters

Hash	table de todas as filiais juntas
Tamanho	da hash table
Numero	de produtos a ir buscar
Flag	para saber se estamos a executar a query 11 ou 12

Returns

Lista de produtos

2.7.3.15 getClientsAndUnitsCLS()

```
int getClientsAndUnitsCLS (
            CLSEARCHER cls,
            char *** clientes,
            int flag )
```

Função que retira os clientes e as unidades da Estrutura de procura "CLSEARCHER".

Parameters

Estrutura	de procura "CLSEARCHER"
Lista	de strings onde sera guardado os clientes
Flag	para determinar o tipo N ou P

Returns

Numero de unidades

2.7.3.16 getCLSearcher()

```
CLSEARCHER getCLSearcher (
            SALES sale,
            char * productID )
```

Função que encontra as clientes que compraram um produto.

Parameters

Estrutura	das vendas
Produto	a procurar

Returns

Estrutura que guarda os clientes divindindo-os em compras N e P

2.7.3.17 getProductsAndUnitsPRS()

```
char** getProductsAndUnitsPRS (
            PRSEARCHER prs,
            int size,
            int qt[] )
```

Função encontra os produtos da estrutura "PRSEARCHER" e coloca-os numa lista de strings.

Parameters

Estrutura	de procura "PRSEARCHER"
Tamanho	da lista

Returns

Lista de produtos

2.7.3.18 initClientProducts()

```
CLPROD initClientProducts ( )
```

Inicialização da Estrutura que relaciona Clientes com Produtos.

Returns

Estrutura que relaciona Clientes com Produtos inicializada

2.7.3.19 initProductsClients()

```
PRODCL initProductsClients ( )
```

Inicialização da Estrutura que relaciona Produtos com Clientes.

Returns

Estrutura que relaciona Produtos com Clientes inicializada

2.7.3.20 initSales()

```
SALES initSales ( )
```

Inicializador da Estrutura das Vendas.

Returns

Estrutura das Vendas

2.7.3.21 mergeBranches()

```
GHashTable* mergeBranches (
            SALES * sales )
```

Função que junta as filiais numa unica hash table.

Parameters

Array	de Estruturas das vendas

Returns

Hash table retornada

2.7.3.22 mergeBranchesAndMonths()

```
GHashTable* mergeBranchesAndMonths (
            SALES * sales,
            char * client )
```

Função que junta as filiais devididas em meses numa unica hash table.

Parameters

Array	de Estruturas das vendas

Returns

Hash table retornada

2.7.3.23 numberOfSales()

```
int numberOfSales (
            PRODCL pc,
            int flag )
```

Função que encontra o numero de vendas feitas.

Parameters

Estrutura	que relaciona produtos com clientes
Flag	para procurar clientes que fizeram compras de tipo N ou P

Returns

Numero de vendas

2.7.3.24 sizeOfProducts()

```
int sizeOfProducts (
            SALES sales,
            char * client,
            int month )
```

Função que calcula o numero de produtos comprados por um cliente.

Parameters

Estrutura	das vendas
Cliente	a ser procurado
Mes	a ser procurado

Returns

numero de produtos comprados

2.7.3.25 sort()

```
int sort (
            const void * one,
            const void * two )
```

Função auxilar que retorna se o segundo parametro é maior do que o primeiro.

Parameters

String	1
String	2

Returns

Resposta da comparaçao de duas strings

2.8 sgv.h File Reference

Modulo de tratamento da Base de Dados.

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>
#include "clients.h"
#include "products.h"
#include "sales.h"
```

Typedefs

- typedef struct sgv * SGV

  Base de Dados.

Functions

- SGV initialize_SGV ()

  Inicializador da Base de Dados.
- void destroctionSGV (SGV sgv)

  Destruiçao da Base de Dados.
- SGV loadSGV (SGV sgv, char *clientsFilePath, char *productsFilePath, char *salesFilePath)

Carrega Bases de Dados (SGV) com os ficheiros.

- void readClients (SGV sgv, char ∗clientsFilePath)

  Carrega Estrutura dos clientes a partir do ficheiro dos clientes.
- void readProducts (SGV sgv, char ∗productsFilePath)

  Carrega Estrutura dos produtos a partir do ficheiro dos produtos.
- void readSales (SGV sgv, char ∗salesFilePath)

  Carrega Estrutura das vendas a partir do ficheiro das vendas.
- void changeTotalMonth (SGV sgv, float price, int units, int month)

  Altera o numero de vendas faturação total por meses.
- int validadeSale (SGV sgv, char ∗∗token)

  Verifica se uma venda é valida.
- int validateProduct (char ∗buffer)

  Verifica se um produto é bem contruido.
- int validatePrice (char ∗price)

  Verifica se um preço é bem contruido.
- int validateUnits (char ∗units)

  Verifica se as unidades sao bem contruidas.
- int validateType (char ∗type)

  Verifica se o tipo de venda é bem contruida.
- int validateClient (char ∗buffer)

  Verifica se um cliente é bem contruido.
- int validateMonth (char ∗month)

  Verifica se o mes da venda é bem contruida.
- int validateBranch (char ∗branch)

  Verifica se a filial da venda é bem contruida.
- PRODUCTS getArrayProductsByLetter (SGV sgv, char letter)

  Função que encontra o array de produtos correspondente a uma letra.
- PRODUCTS ∗ getArrayProducts (SGV sgv)

  Função que encontra o array de produtos.
- SALES ∗ getArraySales (SGV sgv)

  Função que retorna o array de filias.
- SALES getSales (SGV sgv, int branch)

  Função que retorna um Estrutura de Filias.
- CLIENTS getClients (SGV sgv)

  Função que retorna a Estrutura dos Clientes.
- int ∗ getLinesValidatedAndRead (SGV sgv)

  Função que retorna as linhas lidas e as validas da leitura dos files.
- void findSalesAndProfit (SGV sgv, int minMonth, int maxMonth, double res[ ])

  Função que retorna a Estrutura dos Clientes.

2.8.1 Detailed Description

Modulo de tratamento da Base de Dados.

2.8.2 Typedef Documentation

2.8.2.1 SGV

```
typedef struct sgv* SGV
```

Base de Dados.

2.8.3 Function Documentation

2.8.3.1 changeTotalMonth()

```
void changeTotalMonth (
            SGV sgv,
            float price,
            int units,
            int month )
```

Altera o numero de vendas faturação total por meses.

Parameters

Base	de Dados (SGV)
Preço	unitario da vendas
Unidades	da venda
Mes	da venda

2.8.3.2 destroctionSGV()

```
void destroctionSGV (
            SGV sgv )
```

Destruiçao da Base de Dados.

Parameters

Estrutura	a ser destruida

2.8.3.3 findSalesAndProfit()

```
void findSalesAndProfit (
            SGV sgv,
```

```
            int minMonth,
            int maxMonth,
            double res[] )
```

Função que retorna a Estrutura dos Clientes.

Parameters

Estrutura	de Dados (SGV)
Mes	minimo
Mes	maximo
Array	que ira guardar a resposta

2.8.3.4 getArrayProducts()

```
PRODUCTS* getArrayProducts (
            SGV sgv )
```

Função que encontra o array de produtos.

Returns

Array da Estrutura dos produtos

2.8.3.5 getArrayProductsByLetter()

```
PRODUCTS getArrayProductsByLetter (
            SGV sgv,
            char letter )
```

Função que encontra o array de produtos correspondente a uma letra.

Parameters

Estrutura	de Dados (SGV)
Letra	a qual ir corresponde o array

Returns

Estrutura dos produtos

2.8.3.6 getArraySales()

```
SALES* getArraySales (
            SGV sgv )
```

Função que retorna o array de filias.

Parameters

Estrutura	de Dados (SGV)

Returns

Estrutura de filias

2.8.3.7 getClients()

```
CLIENTS getClients (
            SGV sgv )
```

Função que retorna a Estrutura dos Clientes.

Parameters

Estrutura	de Dados (SGV)

Returns

Estrutura dos Clientes

2.8.3.8 getLinesValidatedAndRead()

```
int* getLinesValidatedAndRead (
            SGV sgv )
```

Função que retorna as linhas lidas e as validas da leitura dos files.

Parameters

Estrutura	de Dados (SGV)

Returns

Array com o numero das linhas lidas e as validas da leitura dos files

2.8.3.9 getSales()

```
SALES getSales (
            SGV sgv,
            int branch )
```

Função que retorna um Estrutura de Filias.

Parameters

Estrutura	de Dados (SGV)
Filia	a ser procurada

Returns

Estrutura de filias

2.8.3.10 initialize_SGV()

```
SGV initialize_SGV ( )
```

Inicializador da Base de Dados.

Returns

Estrutura SGV

2.8.3.11 loadSGV()

```
SGV loadSGV (
            SGV sgv,
            char * clientsFilePath,
            char * productsFilePath,
            char * salesFilePath )
```

Carrega Bases de Dados (SGV) com os ficheiros.

Parameters

Ficheiro	dos clientes
Ficheiro	dos produtos
Ficheiro	das vendas

Returns

Bases de Dados (SGV) carregado

2.8.3.12 readClients()

```
void readClients (
            SGV sgv,
            char * clientsFilePath )
```

Carrega Estrutura dos clientes a partir do ficheiro dos clientes.

Parameters

Base	de Dados (SGV)
Ficheiro	dos clientes

2.8.3.13 readProducts()

```
void readProducts (
            SGV sgv,
            char * productsFilePath )
```

Carrega Estrutura dos produtos a partir do ficheiro dos produtos.

Parameters

Base	de Dados (SGV)
Ficheiro	dos produtos

2.8.3.14 readSales()

```
void readSales (
            SGV sgv,
            char * salesFilePath )
```

Carrega Estrutura das vendas a partir do ficheiro das vendas.

Parameters

Base	de Dados (SGV)
Ficheiro	das vendas

2.8.3.15 validadeSale()

```
int validadeSale (
            SGV sgv,
            char ** token )
```

Verifica se uma venda é valida.

Parameters

Base	de Dados (SGV)
Apontador	de strings a serem validadas

2.8.3.16 validateBranch()

```
int validateBranch (
            char * branch )
```

Verifica se a filial da venda é bem contruida.

Parameters

String	a ser validada

2.8.3.17 validateClient()

```
int validateClient (
            char * buffer )
```

Verifica se um cliente é bem contruido.

Parameters

String	a ser validada

2.8.3.18 validateMonth()

```
int validateMonth (
            char * month )
```

Verifica se o mes da venda é bem contruida.

Parameters

String	a ser validada

2.8.3.19 validatePrice()

```
int validatePrice (
            char * price )
```

Verifica se um preço é bem contruido.

Parameters

String	a ser validada

2.8.3.20 validateProduct()

```
int validateProduct (
            char * buffer )
```

Verifica se um produto é bem contruido.

Parameters

String	a ser validada

2.8.3.21 validateType()

```
int validateType (
            char * type )
```

Verifica se o tipo de venda é bem contruida.

Parameters

String	a ser validada

2.8.3.22 validateUnits()

```
int validateUnits (
            char * units )
```

Verifica se as unidades sao bem contruidas.

Parameters

String	a ser validada

Index