**Breve descrição**

O sistema é responsável para guardar dados dos clientes em disco e, durante a execução, armazenar esses mesmos dados na memoria volátil da máquina. Para tanto foi utilizado um vetor dinâmico e também o conhecimento de arquivos em c para tal tarefa.

1. **Arquivo Dados de arquivo.**

De modo geral, as funções desse arquivo retornam:

|  |  |
| --- | --- |
| **Retorno** | **Status** |
| 1 | caso arquivo correto sucesso |
| 0 | caso o arquivo não pode ser aberto |

1.1 **Detalhes:**

Procedimento init void init(Dados\_de\_arquivos\* n\_a), recebe como parâmetro um ponteiro do tipo Dados\_de\_arquivo, esse ponteiro recebe o nome do arquivo que será aberto ("DATA\_clientes.dat”).

Procedimento close void close(Dados\_de\_arquivos\* n\_a), recebe como parâmetro um ponteiro do tipo Dados\_de\_arquivo, fecha o arquivo por meio da função fclose e libera o espaço de memória alocado para o ponteiro por meio da função free.

Função Valida\_arq\_cliente int valida\_arq\_cliente(Dados\_de\_arquivos\* dados\_de\_arquivo), verifica se o arquivo é valido por meio da condicional, retorna 1 caso arquivo correto ou retorna 0 caso o arquivo não possa ser aberto.

A função DADOS\_DE\_ARQUIVO\_grava\_cliente

int DADOS\_DE\_ARQUIVO\_grava\_cliente(Cliente\* cliente\_a\_ser\_gravado, Dados\_de\_arquivos\* dados\_de\_arquivo), recebe como parâmetro um ponteiro para o cliente e outro com os dados a serem gravados e faz a gravação dos dados no arquivo.

O procedimento DADOS\_DE\_ARQUIVO\_exibe\_todos\_os\_clientes

void DADOS\_DE\_ARQUIVO\_exibe\_todos\_os\_clientes(Dados\_de\_arquivos\* dados\_de\_arquivo), recebe um ponteiro do tipo Dados\_de\_arquivos, e exibe na tela os dados que o cliente inseriu no sistema.

A funcao RAM\* DADOS\_DE\_ARQUIVO\_carrega(Dados\_de\_arquivos\* dados\_de\_arquivo) recebe um arquivo do tipo Dados\_de\_arquivo grava os dados do arquivo em um ponteiro do tipo RAM e retorna esse ponteiro.

1. **Arquivo CLIENTE.H**

O procedimento void CLIENTE\_exibe\_cliente(Cliente\* c), recebe um ponteiro do tipo cliente e exibe na tela os dados dos clientes.

A função Cliente\* CLIENTE\_entrada\_de\_cliente() recebe do usuário os dados do cliente que ele deseja cadastrar e retorna um ponteiro do tipo Cliente\*.

A funcao Cliente\* CLIENTE\_copia(Cliente\* cliente), recebe um ponteiro do tipo Cliente\*, armazena os dados dele em um novo ponteiro do tipo cliente, chamado Cliente \*novo e retorna esse novo ponteiro com os dados copiados.

A funcao int valido(Cliente\* cliente) recebe um ponteiro do tipo Cliente\* e verifica se ele é diferente de NULL, ou seja, se é valido, se sim retorna TRUE que vale 1, senão retorna FALSE que vale 0.

O procedimento void CLIENTE\_copy(Cliente\* Destination, Cliente\* Source, int change\_code) recebe dois ponteiros do tipo Cliente\*, um que contem os dados, chamado Source e outro que irá receber os dados, chamado Destination, o change code serve para validar se o codigo sera copiado, se for igual a 1 ele é copiado.

1. **Arquivo Origem.c**

A funcao int menu\_cliente\_exibicao() exibe na tela as opções que o usuario tem para interagir com o programa, recebe um número do usuário de 1 a 7 e retorna esse valor para a main.

A funcao int main() é responsavel pela chamada da funcao que inicia o ponteiro (Dados\_de\_arquivos\* dados) dados do tipo Dados\_de\_arquivo, e ele receberá as informações que estão armazenadas no arquivo. Nela temos um while que contém um switch case, que contém as opções que o cliente pode escolher e de acordo com a escolha executa uma determinada função, pode inserir um usuário(1), alterar(2), consultar(3), excluir(4), listar(5), gravar(6), sair sem salvar(7), e dentro de cada case ocorre a execução de cada uma dessas tarefas.

1. **Arquivo validação.h**

A enumeração enum status faz com que sejam apenas dois os possíveis para a validação da string, toda vez que a função dada for chamada.

|  |  |
| --- | --- |
| **Nome** | **Valor** |
| status\_valido | 1 |
| status\_invalido | 0 |