UNIVERSIDADE PAULISTA – UNIP EaD

Projeto Integrado Multidisciplinar

Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas

RAFAEL JUNGES - 2173421

CAD DIGITAL SUS COVID - 19

São Leopoldo

2021

RAFAEL JUNGES - 2173421

CAD DIGITAL SUS - COVID - 19

Projeto Integrado Multidisciplinar para obtenção do título de tecnólogo em Análise e Desenvolvimento de Sistemas, apresentado à Universidade Paulista – UNIP EaD.

Orientador(a): Professora Vanessa Santos Lessa

São Leopoldo

2021

Resumo

O software demonstrado nesse trabalho tem a função de cadastrar pessoas diagnosticadas com covid-19 em Postos de Saúde e Hospitais do SUS. O software Cad Digital SUS – Covid 19 foi desenvolvido em linguagem C e possui um banco de dados para armazenamento dos dados dos pacientes cadastrados. Durante esse trabalho é apresentado, além do conhecimento de instalação e operação, o conhecimento referente ao funcionamento interno do software, no que tange suas principais funções. Como o código fonte do software é muito extenso com mais de 1300 linhas é inviável coloca-lo diretamente no trabalho, por esse motivo ele segue no Apêndice A – Código Fonte. No que se refere ao código esse trabalho se concentra nos requisitos e na logica de pensamento que levaram a elaboração do código, explicando de maneira conceitual oque esta presente no código. Já sobre a instalação e operação este trabalho explica detalhadamente com faze-lo, deixando claro os procedimentos necessários para que o usuário consiga realizar corretamente as funções para o qual ele foi desenvolvido.

Palavras-chave: Linguagem C. Compilador. Cadastramento. Lógica. Programação.

Abstract

The software demonstrated in this work has the function of registering people diagnosed with covid-19 in SUS Health Posts and Hospitals. The Cad Digital SUS – Covid 19 software was developed in C language and has a database for storing the data of registered patients. During this work, in addition to the knowledge of installation and operation, the knowledge regarding the internal workings of the software, regarding its main functions, is presented. As the software source code is very extensive, with more than 1300 lines, it is not feasible to put it directly at work, for this reason it follows in Appendix A - Source Code. With regard to code, this work focuses on the requirements and the logic of thought that led to the development of the code, explaining in a conceptual way what is present in the code. As for the installation and operation, this work explains in detail how to do it, making clear the necessary procedures for the user to be able to correctly perform the functions for which it was developed.

Keywords: Language C. Compiler. Registration. Logic. Schedule.

SUMÁRIO:

1. INTRODUÇÃO

Com o avanço da pandemia de covid-19 pelo mundo, os países foram forçados a buscar diversas soluções internas para tentar conter a disseminação do vírus. O controle da transmissão é fundamental para proteger os sistemas de saúde e evitar a superlotação e colapso. Umas das maneiras de entender a extensão do contagio e cadastrar os pacientes diagnosticados com coivd-19 para analise de dados e direcionamento das politicas públicas.

Esse cadastramento deve ser feito em hospitais e postos de saúde por pessoal habilitado, para realizar essa tarefa, este projeto desenvolveu um software de cadastramento de pacientes diagnosticados com coivd-19. Este software além de acelerar o processo de cadastramento, ele registra os dados dos pacientes em um banco de permitindo futuras consultas pela equipe médica e caso o paciente seja maior de 65 anos de idade produz um arquivo em separado, informando o Cep e a idade do paciente para envio Secretaria da Saúde da cidade.

2. INSTALAÇÃO DO SOFTWARE

A instalação do software Cad Digital SUS – Covid 19 é bastante simples, basta copiar o executável “CAD_DIGITAL_SUS_COVID-19.exe” para o diretório onde se deseja deixá-lo e pronto, já está instalado e pronto para ser executado.

A primeira vez que ele for executado ele criara automaticamente dois diretórios no mesmo local onde estiver o executável “CAD_DIGITAL_SUS_COVID-19.exe”. Esses diretórios são:

- Pacientes – Esta pasta funciona como um banco de dados dos pacientes recebendo automaticamente, um arquivo por paciente. Cada arquivo contém todos os dados do paciente cadastrados no sistema e o nome desse arquivo é o nome do paciente adicionado a extensão .txt;

- P_Grupo_de_Risco – Esta pasta recebe um arquivo que contêm o CEP e a idade de todos os pacientes cadastrados que possuírem idade superior a 65 anos, ou seja pacientes do grupo de risco. Esse arquivo é diário, ou seja, ele registra em um só arquivo os dados de CEP e idade de todos os pacientes do grupo de risco daquele dia. Ao fim do dia o arquivo é enviado a Secretaria da Saúde da cidade com o nome, “Pacientes do grupo de risco do dia DD-MM-AAAA” onde DD-MM-AAAA é o dia atual.

3. OPERAÇÃO DO SOFTWARE

Para iniciar a operação com o software, execute o executável “CAD_DIGITAL _SUS_COVID-19.exe”. A primeira tela que irá aparecer será a tela de solicitação de Usuário e senha, use:

- Usuário: Digite “Unip” e pressione “Enter”, para confirmar usuário;

- Senha: Digite “VacinasSalvam” e pressione “Enter”, para confirmar senha;

Tela inicial, solicitação de Usuário e Senha:

Imagem 1 – Print de tela, da tela inicial do software.

3.1 Menu Principal

A tela de “Menu Principal” permite ao usuário do sistema efetuar todas as operações para o qual o software foi desenvolvido.

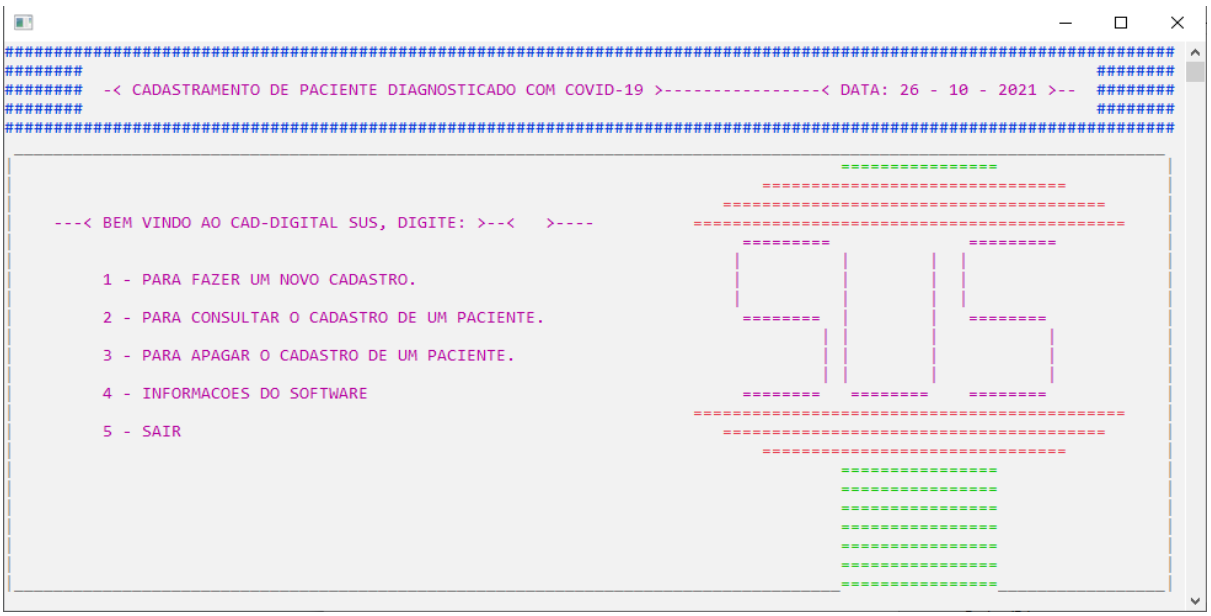

Imagem 2 – Print de tela, da tela de Menu Principal do software.

As operações que podem ser feitas nesse menu são:

- Efetuar o cadastramento de um paciente;
- Consultar o cadastro de um paciente;
- Apagar o cadastro de um paciente;
- Informações do software;
- Sair

3.2 Criando um cadastro de paciente

Para criar um cadastro, no Menu Principal tecle “1”. Aparecerá a primeira tela de cadastro de pacientes, a tela de cadastro de dados pessoais:

Imagem 3 – Print de tela, da tela de Cadastro de Dados Pessoais do software.

Preencha atentamente cada campo e tecle “Enter” ao final. Para evitar cadastro de dados errados, o sistema exige que sejam preenchidos os dados conforme o modelo de dados esperado. Caso contrário o sistema informa o erro de formato e solicita uma nova digitação. Se o formato do dado inserido no campo estiver correto, o sistema passa automaticamente para o próximo campo.

Se todos os campos foram digitados corretamente, o sistema oferece ao operador do software uma verificação dos dados digitados antes de prosseguir para o próximo cadastro.

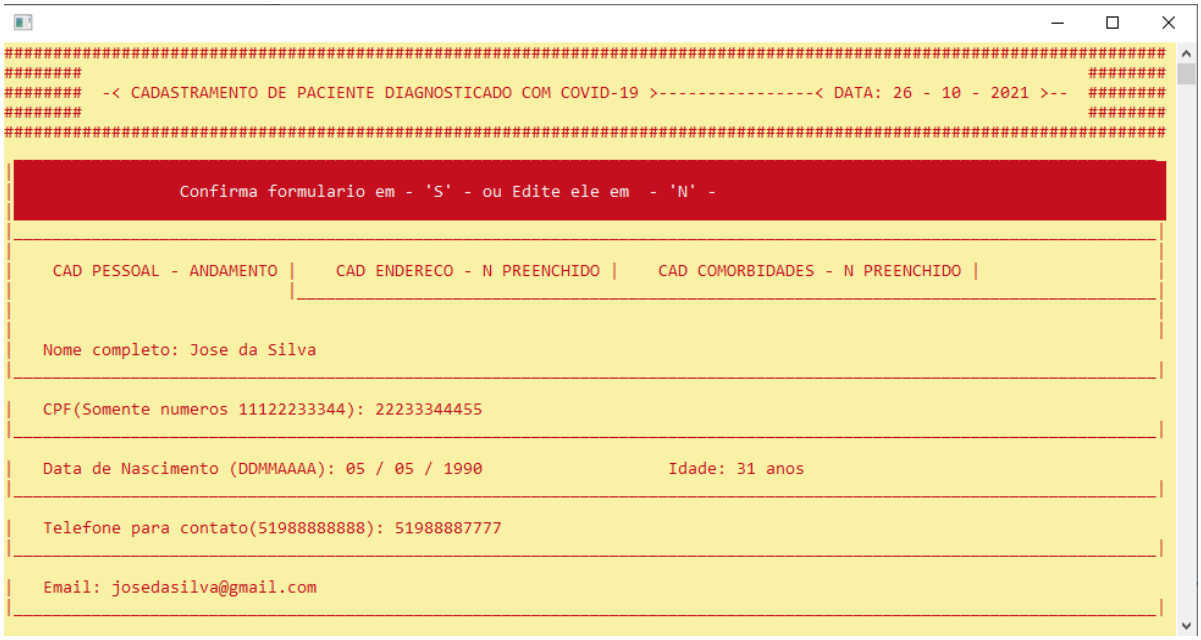

Imagem 4 – Print de tela, da tela do Cadastro de Dados Pessoais do software.

Se todos os dados estiverem preenchidos corretamente o operador deve:

- Teclar “S” para salvar os dados e ir para o próximo cadastro;
- Teclar “N” para refazer o cadastro em tela.

Os demais cadastros o cadastramento de endereço e de comorbidades, seguem a mesma logica, ou seja:

- Preencha os campos e tecle “Enter” para confirmar e ir para o próximo campo;
- Se o dado fornecido no campo for fora do padrão esperado o sistema mostra um erro, e solicita nova digitação;
- Todos os campos são obrigatórios;
- Ao final do cadastro antes de seguir para a próxima tela é necessário confirmar os dados e teclar “S” para prosseguir ou “N” para refazer;

Cadastro de endereço devidamente preenchido:

Imagem 5 – Print de tela, da tela do Cadastro de Endereço do software.

Confirmação de salvamento dos dados de endereço:

Imagem 6 – Print de tela, da tela do Cadastro de Endereço do software, confirmação de cadastro.

Cadastro de comorbidades devidamente preenchido:

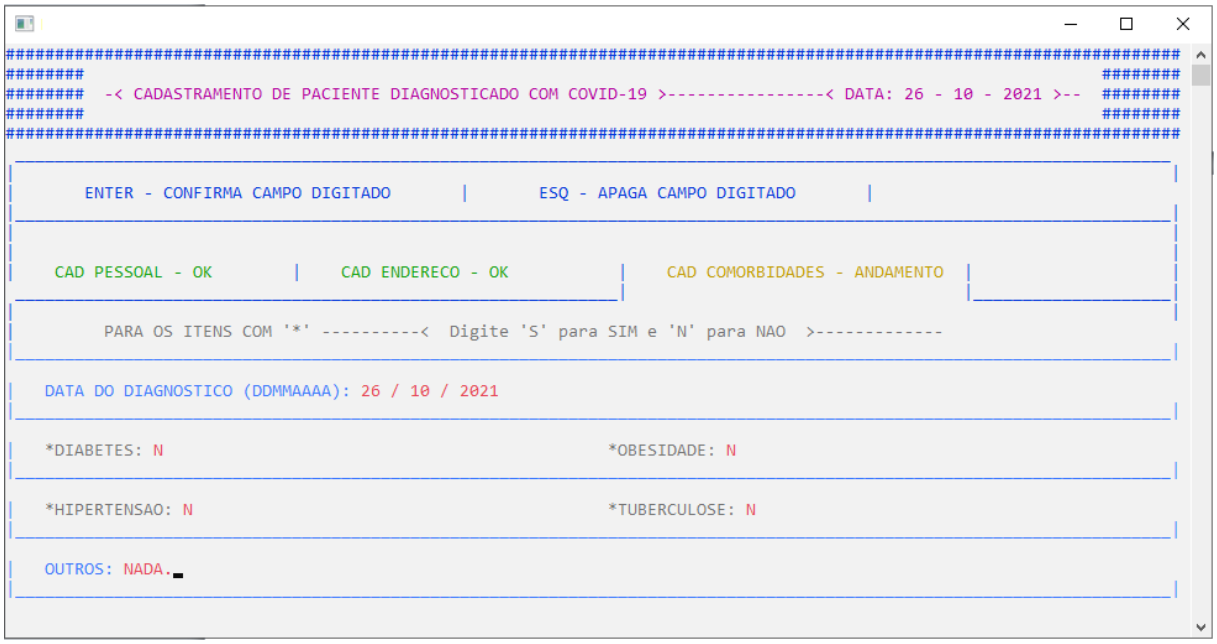

Imagem 7 – Print de tela, da tela do Cadastro comorbidades do software.

Confirmação de salvamento dos dados de comorbidade:

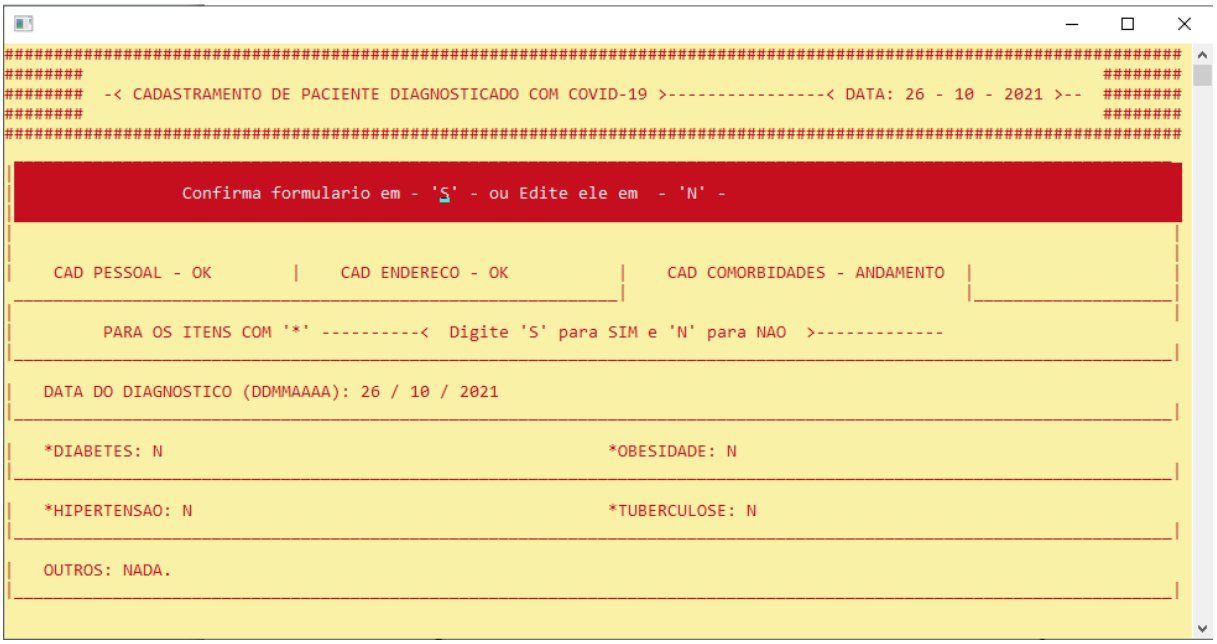

Imagem 8 – Print de tela, da tela do Cadastro de Comorbidades do software, confirmação de cadastro.

Após a finalização do último cadastro, e teclando “S” para salvar os dados, o sistema imprime o cadastro como um todo na tela. Nessa tela também é mostrado se o paciente é ou não do grupo de risco, ou seja, se possui mais de 65 anos de idade.

Imagem 8 – Print de tela, da tela de finalização do cadastro.

Nesse ponto é possível fazer cinco ações diferentes, a depender do desejo do operador do sistema, digitando:

- 1 – Para salvar o cadastro – Nesse ponto o salvamento é definitivo, ou seja, o cadastro ficara disponível para consulta ano banco de dados e não poderá mais ser editado, confirmando o salvamento retorna ao Menu Principal;
- 2 – Para editar Cadastro de dados pessoais – É chamado novamente a tela de Cadastro de Dados Pessoais para edição de dados e ao final retorna para a tela atual;
- 3 – Para editar Cadastro de Endereço – É chamado novamente a tela de Cadastro de Endereço para edição de dados e ao final retorna para a tela atual;

- 4 – Para editar Cadastro de Comorbidades – É chamado novamente a tela de Cadastro de Comorbidades para edição de dados e ao final retorna para tela atual;
- 5 – Para apagar todo o cadastro dessa fixa – Apaga todos os dados salvos para essa fixa, e retorna para o Menu Principal;

3.3 Consultando o cadastro de um paciente

Para consultar o cadastro de um paciente, no Menu Principal e tecle “2”, aparecera a tela de pesquisa, e nela você deve digitar o nome completo do paciente, do mesmo modo como foi cadastrado:

Imagem 9 – Print de tela, da tela de consulta de cadastro.

Após digitar o nome corretamente o software pesquisa no banco de dados se existe o cadastro desejado. Caso exista o cadastro, o sistema imprime na tela todos os dados do paciente para a consulta, caso não exista o cadastro ou alguma palavra tenha sido digitada errada e o sistema no localiza o arquivo, o sistema imprime na tela “Cadastro não Localizado” e retorna ao menu inicial.

Segue abaixo exemplo de busca bem sucedida:

Imagem 10 – Print de tela, da tela dos dados de um cadastro.

3.3 Apagando o cadastro de um paciente

Para apagar o cadastro de um paciente, no Menu Principal e tecle “3”, aparecera a tela de pesquisa, e nela você deve digitar o nome completo do paciente, do mesmo modo como foi cadastrado:

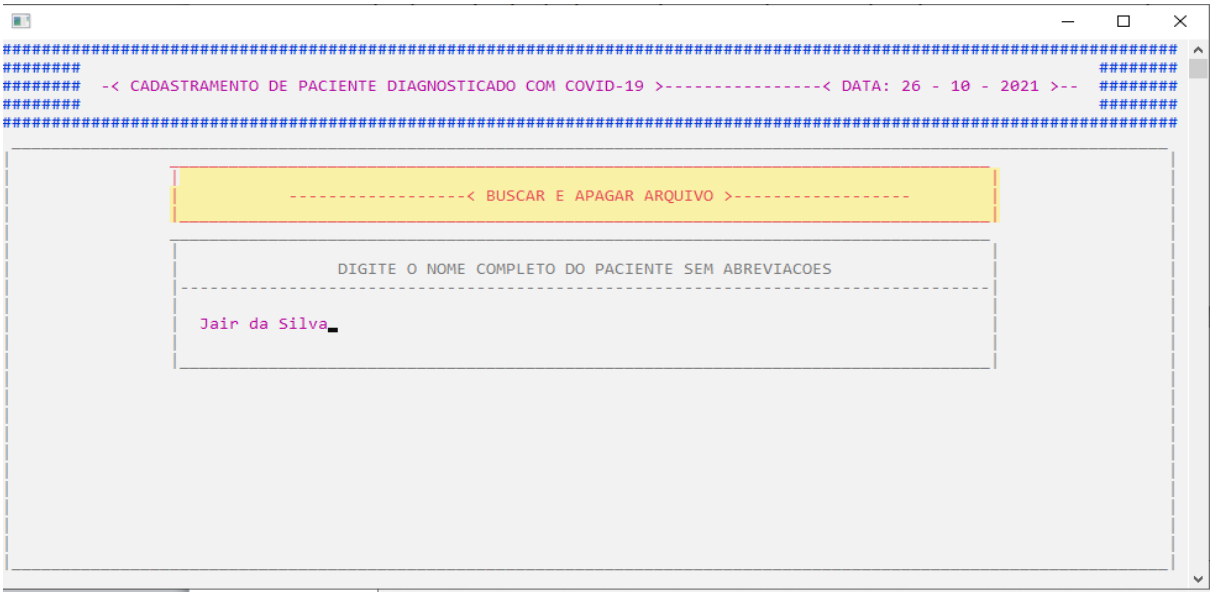

Imagem 11 – Print de tela, da tela de consulta para apagar um cadastro.

Após digitar o nome corretamente o software pesquisa no banco de dados se existe o cadastro desejado. Caso exista o cadastro, o sistema apaga o cadastro e retorna ao Menu Inicial, caso contrário o sistema imprime na tela “Cadastro não Localizado” e retorna ao menu inicial.

3.4 Informações do Software

Para visualizar informações do sistema, no Menu Principal e tecle “4”, e aparecera informações do software e do desenvolvedor.

3.5 Encerrado o Software

Para encerrar o software, no Menu Principal e tecle “5”, e se encerrara a execução do programa.

Importante ressaltar que mesmo fechando o programa ou desligando o computador, como cada cadastro esta salvo em um banco de dados na pasta “Pacientes”, todos esses dados podem ser acessados posteriormente.

4. DESENVOLVENDO O SOFTWARE: Cad Digital SUS – Covid 19

O software Cad Digital SUS – Covid 19 foi desenvolvido em linguagem C com o auxílio do compilador de código *Code::Blocks,* que permite ao programador criar o código realizar testes, visualizar erros, corrigir e melhorar o código. Para realizar a codificação do software foi necessário analisar os objetivos do projeto, que estão presentes no manual do PIM, são eles:

- O software terá a função de cadastrar pacientes diagnosticados com covid-19;
- Realizar *Login* no sistema para operar o software, para o *login* o software deve solicitar usuário e senha.
- Cadastrar os dados pessoais, dados de endereço e dados de comorbidades associadas ao paciente;
- Ao final de cada cadastro os dados do paciente devem ser salvos em arquivo para consultas posteriores;

- Ao final do cadastro se o paciente for maior de 65 anos de idade o software de produzir um arquivo de texto para envio a Secreteria da Saúde da Cidade.

4.1 Código fonte

Devido ao tamanho do código fonte, contendo exatamente 1389 linhas e ao fato das funções principais também serem bastante grandes e haver muitas Interrelações entre elas, o código fonte, que está todo comentado, estará no Apêndice A – Código Fonte.

4.2 Desenvolvendo a validação de entrada – Login

Como foi solicitado pelo cliente o software solicita validação de usuário e senha, para isso, na tela inicial é solicitado a digitação de usuário, o software captura o que foi digitado no teclado e converte esse dado numa *string* que é após pressionado “Enter” armazena o valor digitado numa variável auxiliar, repete o procedimento anterior para senha salvando em outra variável auxiliar. O valor dessas variáveis é comparado com o valor de referência salvo nas variáveis “user” e “senha”, se estiver exatamente igual o *login* e aprovado e o programa segue para a próxima tela, caso contrário é solicitada nova digitação.

4.3 Criando uma estrutura de dados do paciente

Cada paciente é igual quanto aos dados a serem cadastrados, por esse motivo a melhor maneira de criar a estrutura de dados dele, é utilizando a função “*struct*” que permite uma estruturação centralizada de tudo que se refere ao paciente. Como há a necessidade de cálculo da idade do paciente foi necessário a criação de uma estrutura de dados que capturasse a data real do dia atual do sistema operacional, para isso, além de uma função especifica criada para esse fim foi criado também uma “*struct*”, para estruturar melhor esses dados.

4.4 Garantido a qualidade dos dados adquiridos

Para garantir a qualidade dos dados inseridos no software é fundamental garantir que os campos de entrada de dados recebam os dados corretos, por exemplo, não permitir que campos vazios sejam aceitos, ou que haja letras onde somente deveria haver números. Para resolver isso, os dados somente são aceitos se passarem no teste específico do campo em preenchimento. Todos os campos são requeridos, e caso o operador tente passar por ele, com ele fora do padrão solicitado ou sem digitação alguma o software solicita nova digitação.

Segue abaixo uma tabela demonstrando os testes feitos para campos de números:

QUADRO DE TESTES PARA ACEITAÇÃO DO DADO NO CAMPO REQUERIDO				
Dados	Caracteres	Numeros	Quantidade exata de digitos	Especificidade do dado
Nome, Rua, Número, Bairro, Cidade, Estado, Comorbidades outras	SIM	NÃO	não tem	- nada -
Telefone	NÃO	SIM	11	55988887777
Cpf	NÃO	SIM	11	22255588899
Cep	NÃO	SIM	8	99555222
Data de Nascimento	NÃO	SIM	8	Dia-01 a 31 --- Mês 01 a 12 ---- Ano 1900 a 2021 DD/MM/AAAA
Data de Diagnostico	NÃO	SIM	8	Dia-01 a 31 --- Mês 01 a 12 ---- Ano 1900 a 2021 DD/MM/AAAA

Quadro 01 – Quadro de testes para cada campos de números

Segue abaixo uma tabela demonstrando os testes feitos para campos de especiais:

QUADRO DE TESTES PARA ACEITAÇÃO DO DADO NO CAMPO REQUERIDO				
Dados	Caracteres	Numeros	Quantidade exata de digitos	Especificidade do dado
email	SIM	SIM	não tem	- nada -
(Endereço) Numero:	NÃO	SIM	não tem	No minimo 1 digito
Diabetes	SIM	NÃO	1	S ou N
Obesidade	SIM	NÃO	1	S ou N
Hipertensao	SIM	NÃO	1	S ou N
Tuberculose	SIM	NÃO	1	S ou N

Quadro 03 – Quadro de testes para cada campos de especiais

4,5 Salvando os dados e criando o arquivo de paciente

Durante o processo de digitação após a validação de cada campo, o valor do campo é instantaneamente copiado para o campo apropriado na estrutura de dados do paciente, por exemplo, após o operador do software ter digitado o nome do paciente, e o nome ter sido aprovado no teste, o conteúdo dessa digitação é copiado para a variável nome do paciente. Esse procedimento de digitação, validação e copia para a estrutura de dados, segue para todos os campos solicitados nas telas de cadastros. Quando durante o processo é solicitado edição de uma tela já editada os valores novos sobrescrevem os antigos nos campos editados.

Ao final da digitação toda a estrutura de dados do paciente estará completa, o operador do software recebera na tela uma visão completa do cadastro para avaliação, se estiver tudo certo o operador aceita o cadastro e nesse momento o software cria um arquivo na pasta “Pacientes” como o nome “NomedoPaciente.txt”, esse arquivo recebe ordenadamente, todos os dados do paciente. Cada dado fica numa linha, na ordem em que foram digitados no software, esse formato de exportação é importante porque permite uma fácil importação desses dados posteriormente.

Print de tela dos dados contidos no cadastro de teste de “Jose da Silva”, salvo na pasta local “Pacientes” junto ao executável Cad Digital SUS – Covid 19. exe

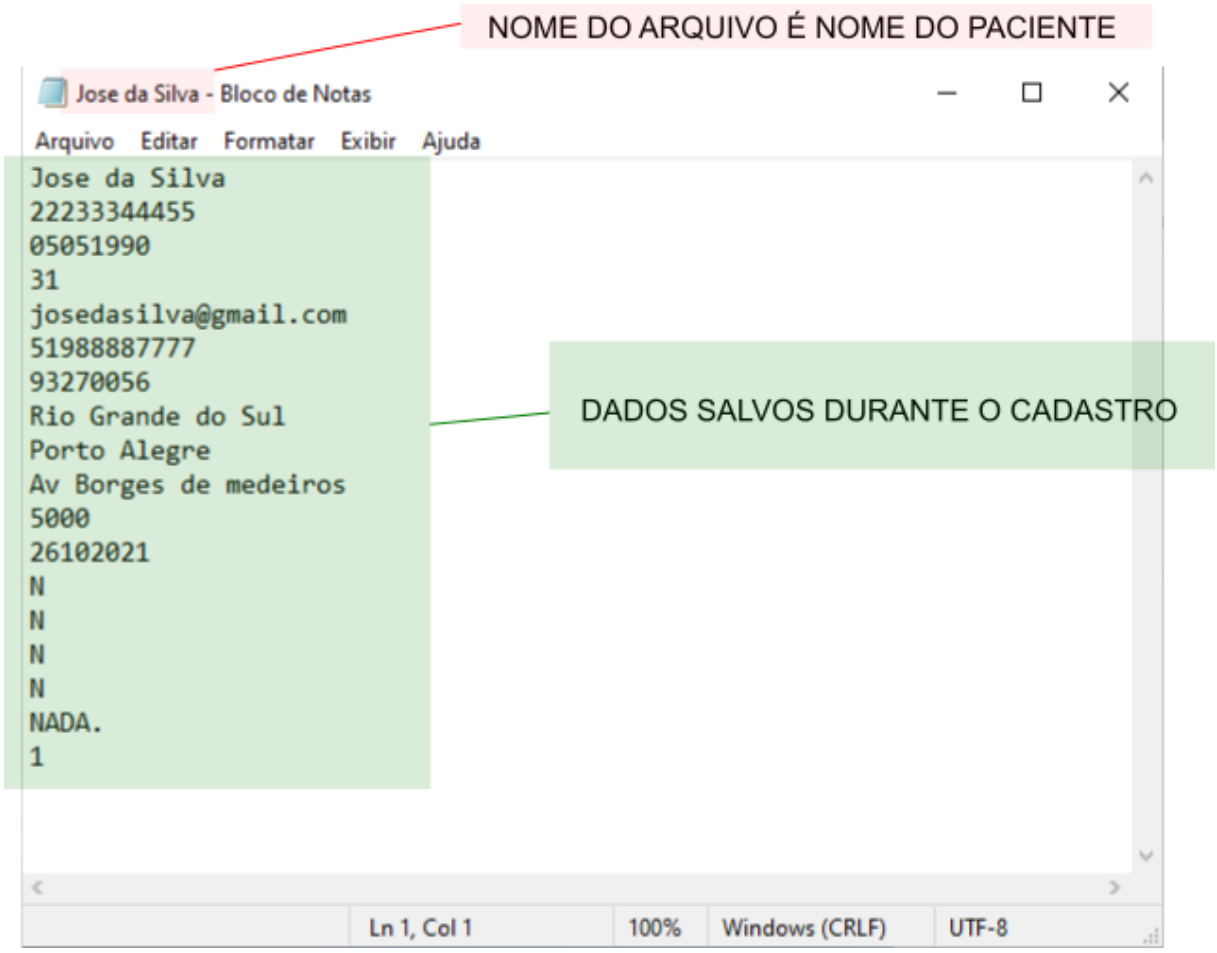

Imagem 12 – *Print* de tela, de arquivo de paciente.

4.6 Consultando os dados de um paciente

Como o salvamento dos dados esta centralizado na pasta “Paciente”, o software procura somente nessa pasta os arquivos de paciente. Caso a busca seja bem sucedida, ou seja, o nome do arquivo solicitado na busca e o nome do arquivo real presente na pasta “Paciente”, o programa abre o arquivo e importa o valor contido em cada linha até o final do arquivo, fechando após a leitura. O valor de cada linha é colocado numa variável auxiliar diferente, após a captura de todos o dados, o software organiza esses dados adiciona texto de tela e imprime na tela para visualização do operador do software. Caso o software não encontre o arquivo

pesquisado, é impresso na tela a informação de que o arquivo não foi encontrado e é o software retorna ao Menu Principal.

4.7 Apagando os dados de um paciente

A logica para o apagamento de arquivo de paciente, é bastante parecida com a de busca para consulta com a diferença, de que, caso o arquivo seja encontrado ele não será lido pelo software a única função a ser executada será a de deletar o arquivo. Caso o software não encontre o arquivo pesquisado, é impresso na tela a informação de que o arquivo não foi encontrado e é o software retorna ao Menu Principal.

5. CONCLUSÃO

Conclui-se na base no desenvolvimento desse trabalho, que a ferramenta de desenvolvimento *Code::Blocks* revelou-se bastante robusta e completa, eficiente desde a analise de erros do código, compilação e geração do executável, tudo em um único ambiente de desenvolvimento. Pode ser avaliado também que é possível desenvolver um software robusto em linguagem C, capaz de realizar de maneira simples e rápida o cadastramento de pessoas para situações em geral, no caso especifico o software cadastra pessoas diagnosticadas com covid 19, porem com pequenas alterações seria possível, cadastrar pessoas para diversos outro fins, como pequenas lojas e comércios em geral.

REFERÊNCIAS

BIBLIOGRÁFICAS:

OLIVEIRA, Luis Rodolfo Marques de. **Princípios de Sistemas de Informação** - São Paulo: Editora Sol. 2019. 176 p.

IVO, Olavo. **Linguagens e técnicas de programação** - São Paulo: Editora Sol. 2014. 488 p.

SITES:

Parâmetro Em: System(“Color”, Var) Na Linguagem C?. **clubedohardware**, 2 de abril de 2017. Disponível em: https://www.clubedohardware.com.br/topic/1222125-par%C3%A2metro-em-system%E2%80%9Ccolor%E2%80%9D-var-na-linguagem-c/ . Acesso em: 15 de outubro de 2021.

Como obter os valores de data e hora em um programa em C?. **ti-enxame.com**, de 15 abril de 2017. Disponível em: https://www.ti-enxame.com/pt/c/como-obter-os-valores-de-data-e-hora-em-um-programa-em-c/967078888/
. Acesso em: 15 de outubro de 2021.

Arquivos em C. **http://linguagemc.com.br/**. Disponível em: http://linguagemc.com.br/arquivos-em-c-categoria-usando-arquivos/. Acesso em: 21 de outubro de 2021.

APÊNDICE - A CÓDIGO FONTE COMENTADO

```
#include <stdio.h>
#include <stdlib.h>
#include <windows.h>
#include <conio.h>
#include <time.h>

////////////////////////////////// estruturas iniciais de data e paciente ///////////////
    struct tmx {

    int tm_mday; //dia do mês de 1 a 31
    int tm_mon; //representa os meses do ano de 0 a 11
    int tm_year; //representa o ano a partir de 1900
    };
    struct tm *data;// estrutura de captura de data do sistema;
   struct Paciente{ // Estrutura de criacao do paciente
    int idade,risco;
    char nome[500],cpf[500],datanasc[500],datadiagn[500],
         email[500],telefone[500],
         cidade[500],estado[500],rua[500],cep[500],numero[500],
         comob1[10],comob2[10],comob3[10],comob4[10],comob5[50000];
     };
  struct Paciente P;

////////////////////////////// Variaveis globais ////////////////////

   int j,cads,diaReal,mesReal,anoReal,tecla,idade,fluxo;
   char cadsCH[1],testaString[500],testoOk[500],ichar;

    int main() //////////principal funcao do software
    {
     CreateDirectory("Pacientes",NULL); ////// cria diretorio qeu funcionara como bando de dados dos
pacientes
     CreateDirectory("P_Grupo_de_Risco",NULL);//// cria diretorio que recebera os pacientes de risco
     Hoje();////////////// chama a funcao que captura a data e hora do computador
     system("color b1");/////deine a cor geral de fundo em branco e textos em azul
     loginSenha(); //// Chama a funcao de entrada login e senha
     return 0;
    }
    void Hoje(){  // estrutura de captura de data do sistema;
    time_t segundos;
    time(&segundos);
    data = localtime(&segundos);
    diaReal = data->tm_mday;
    mesReal = data->tm_mon+1;
    anoReal = data->tm_year+1900;
    }
    void loginSenha(){ ///// Funcao Login e senha: desenha a tela de login e senha e ///// solicita login
e senha e verifica se esta certo
        char user[50]="Unip";
        char senha[50]= "VacinasSalvam";
        char testeUser[50],testSenha[50];
snh:    system("cls");
        cabecalho();
        margens();
        gotoxy(35,10);
        printf("_____");
        gotoxy(35,11);
        printf("|                                          |");
        gotoxy(35,12);
        printf("|                                          |");
        gotoxy(35,13);
        printf("|------------------------------------------|");
        gotoxy(35,14);
        printf("|                                          |");
        gotoxy(35,15);
        printf("|                                          |");
```

```
65          gotoxy(35,16);
66          printf("|                                                |");
67          gotoxy(35,17);
68          printf("|________________________________________________|");
69          gotoxy(43,12);
70          printf("DIGITE SEU LOGIN PARA ACESSAR");
71          gotoxy(38,14);
72          printf("USUARIO:");
73          gotoxy(38,16);
74          printf("SENHA:");
75          gotoxy(10,2);
76          printf("\033[35m\033[1m-< CADASTRAMENTO DE PACIENTE DIAGNOSTICADO COM COVID-19 >----------------<
DATA: %d - %d - %d >--\033[34m\n",diaReal, mesReal ,anoReal);
77          gotoxy(38,20);
78          printf("\033[35m\033[1m * DIGITE O USUARIO E TECLE ----'ENTER'----. \033[34m");
79          gotoxy(38,22);
80          printf("\033[35m\033[1m * DIGITE O SENHA E TECLE ----'ENTER'----. \033[34m");
81          fflush(stdin);
82          gotoxy(47,14);
83          gets(testeUser);
84          fflush(stdin);
85          gotoxy(45,16);
86          gets(testSenha);
87          if((strcmp(user,testeUser)==0)&&(strcmp(senha,testSenha)==0))
88              MenuPrincipal();//// se estiver correto vai para o menu principal;
89          else{ // se nao, mostra que esta incorreto e solicita nova digitação
90              for(j=0;j<5;j++)
91              {
92              gotoxy(38,14);
93              printf("                                        ");
94              gotoxy(38,16);
95              printf("                                        ");
96              gotoxy(38,14);
97              Sleep(500);
98              printf("       USUARIO OU SENHA INCORRETOS");
99              gotoxy(38,16);
100             printf("          TENTE NOVAMENTE");
101             Sleep(500);
102             }
103             goto snh;
104         }
105     }
106     int MenuPrincipal(){ // menu principal desenha a tela e fornece as opcoes de acao ao operador
107     cabecalho();
108     printf("\033[1;90m");
109     margens();
110     gotoxy(75,10);
111     printf("\033[35m\033[1m=========              =========");
112     gotoxy(74,11);
113     printf("|          |        |  |");
114     gotoxy(74,12);
115     printf("|          |        |  |");
116     gotoxy(74,13);
117     printf("|          |        |  |");
118     gotoxy(75,14);
119     printf("========  |        |   ========");
120     gotoxy(83,15);
121     printf("| |        |           |");
122     gotoxy(83,16);
123     printf("| |        |           |");
124     gotoxy(83,17);
125     printf("| |        |           |");
126     gotoxy(75,18);
127     printf("========   ========    ========");
128     gotoxy( 70,19);
129     printf("\033[31m========================================== ");
```

```
130       gotoxy( 73,20);
131       printf("====================================");
132       gotoxy( 77,21);
133       printf("==============================");
134       gotoxy( 85,22);
135       printf("\033[1;92m================");
136       gotoxy( 85,23);
137       printf("================");
138       gotoxy( 85,24);
139       printf("================");
140       gotoxy( 85,25);
141       printf("================");
142       gotoxy( 85,26);
143       printf("================");
144       gotoxy( 85,27);
145       printf("================");
146       gotoxy( 85,28);
147       printf("================\033[34m");
148       gotoxy( 70,9);
149       printf("\033[31m========================================== ");
150       gotoxy( 73,8);
151       printf("====================================");
152       gotoxy( 77,7);
153       printf("==============================");
154       gotoxy( 85,6);
155       printf("\033[1;92m================\033[34m");
156       gotoxy( 5,9);
157       printf("\033[1;95m---< BEM VINDO AO CAD-DIGITAL SUS, DIGITE: >--<   >----");
158       gotoxy( 10,12);
159       printf("1 - PARA FAZER UM NOVO CADASTRO.");
160       gotoxy( 10,14);
161       printf("2 - PARA CONSULTAR O CADASTRO DE UM PACIENTE.");
162       gotoxy( 10,16);
163       printf("3 - PARA APAGAR O CADASTRO DE UM PACIENTE.");
164       gotoxy( 10,18);
165       printf("4 - INFORMACOES DO SOFTWARE");
166       gotoxy( 10,20);
167       printf("5 - SAIR");
168       gotoxy( 43,10);
169       gotoxy(10,2);
170       printf("\033[35m\033[1m-< CADASTRAMENTO DE PACIENTE DIAGNOSTICADO COM COVID-19 >----------------<
DATA: %d - %d - %d >--\033[34m\n",diaReal, mesReal ,anoReal);
171       gotoxy(53,9);
172       tecla=getch();
173       fflush(stdin);
174       switch(tecla){ // Verifica a tecla prescionada e direciona para a funcao requerida pelo operador
175       case 49: // digito 1
176          fluxo=0;
177          cadastraDadosPessoais();
178          break;
179       case 50: // digito 2
180          capturaDados();
181          break;
182       case 51: // digito 3
183         deleletarCadastro();
184          return 0;
185          break;
186      case 52: // digito 4
187         InfoSystema();
188          return 0;
189          break;
190      case 53: // digito 5
191          cabecalho();
192          printf("\033[1;90m");
193          margens();
194          gotoxy(10,2);
```

```
195          printf("\033[35m\033[1m-< CADASTRAMENTO DE PACIENTE DIAGNOSTICADO COM COVID-19 >----------------<
DATA: %d - %d - %d >--\033[34m\n",diaReal, mesReal ,anoReal);
196          gotoxy(10,10);
197          printf("\033[35m\033[1m VOLTE SEMPRE!");
198          gotoxy(10,26);
199          exit(0);
200          return 0;
201          break;
202       default:
203        system("cls");
204        printf("\n\n\n\n\n\n                                        Caracter invalido!, tente novamente
\n\n\n\n\n");
205        system("pause");
206        MenuPrincipal();
207        break;
208       }
209       return 0;
210    }
211       int InfoSystema(){ // desenha na tela a infos do software
212
213       sis: cabecalho();
214       printf("\033[1;90m");
215       margens();
216       gotoxy(75,10);
217       printf("\033[35m\033[1m=========              =========");
218       gotoxy(74,11);
219       printf("|          |        |  |");
220       gotoxy(74,12);
221       printf("|          |        |  |");
222       gotoxy(74,13);
223       printf("|          |        |  |");
224       gotoxy(75,14);
225       printf("========  |        |   ========");
226       gotoxy(83,15);
227       printf("| |        |          |");
228       gotoxy(83,16);
229       printf("| |        |          |");
230       gotoxy(83,17);
231       printf("| |        |          |");
232       gotoxy(75,18);
233       printf("========   ========    ========");
234       gotoxy( 70,19);
235       printf("\033[31m========================================== ");
236       gotoxy( 73,20);
237       printf("====================================");
238       gotoxy( 77,21);
239       printf("==============================");
240       gotoxy( 85,22);
241       printf("\033[1;92m================");
242       gotoxy( 85,23);
243       printf("================");
244       gotoxy( 85,24);
245       printf("================");
246       gotoxy( 85,25);
247       printf("================");
248       gotoxy( 85,26);
249       printf("================");
250       gotoxy( 85,27);
251       printf("================");
252       gotoxy( 85,28);
253       printf("================\033[34m");
254       gotoxy( 70,9);
255       printf("\033[31m========================================== ");
256       gotoxy( 73,8);
257       printf("====================================");
258       gotoxy( 77,7);
```

```
259       printf("==============================");
260       gotoxy( 85,6);
261       printf("\033[1;92m================\033[34m");
262       gotoxy( 5,10);
263       printf("\033[35m\033[1mUNIVERSIDADE PAULISTA UNIP EaD");
264       gotoxy( 5,12);
265       printf("CURSO SUP. DE TEC. EM ANALISE E DESENVOLVIMENTO DE SISTEMAS");
266       gotoxy( 5,14);
267       printf("PROJETO INTEGRADO MULTIDISCIPLINAR - PIM IV");
268       gotoxy( 5,16);
269       printf("ALUNO: RAFAEL JUNGES");
270       gotoxy( 5,18);
271       printf("RA: 2173421");
272       gotoxy(10,2);
273       printf("\033[35m\033[1m-< CADASTRAMENTO DE PACIENTE DIAGNOSTICADO COM COVID-19 >----------------<
DATA: %d - %d - %d >--\033[34m\n",diaReal, mesReal ,anoReal);
274       gotoxy(5,24);
275       system("pause");
276       MenuPrincipal();
277      return 0;
278    }
279      void gotoxy(int x, int y){ // importante funcao q permite navegar com o cursor dentro da tela
280      COORD c = {x,y};
281      SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE),c);
282      }
283      void consultar(){ //desenha tela de consulta
284       system("cls");
285       cabecalho();
286       printf("\033[1;90m");
287       margens();
288          gotoxy(17,10);
289          printf("_____________________________________________________________________________");
290          gotoxy(17,11);
291          printf("|                                                                            |");
292          gotoxy(17,12);
293          printf("|                                                                            |");
294          gotoxy(17,13);
295          printf("|----------------------------------------------------------------------------|");
296          gotoxy(17,14);
297          printf("|                                                                            |");
298          gotoxy(17,15);
299          printf("|                                                                            |");
300          gotoxy(17,16);
301          printf("|                                                                            |");
302          gotoxy(17,17);
303          printf("|____________________________________________________________________________|");
304          gotoxy(34,12);
305          printf("DIGITE O NOME COMPLETO DO PACIENTE SEM ABREVIACOES");
306      }
307      void deleletarCadastro(){// desenha na tela a tela de deletar busca o arquivo e deleta se encontrado
308       char extencao[10] = ".txt"; // extensao do arquivo a ser gerado
309       char caminho[500]="Pacientes/"; // pasta local onde estao os arquivo dos pacientes
310       char pesquisa[500];
311       char texto[1000];
312       FILE *arquivo;
313       int w;
314          consultar();
315          gotoxy(17,6);
316          printf(
"\033[31m_____________________________________________________________________________");
317          gotoxy(17,7);
318          printf("|\033[31m\033[1;103m
         |");
319          gotoxy(17,8);
320          printf("|                                                                            |");
321          gotoxy(17,9);
```

```
322          printf("|_____________________________________________________________________________|");
323          gotoxy(29,8);
324          printf("\033[31m\033[1m-----------------< BUSCAR E APAGAR ARQUIVO
>-----------------\033[34m\033[1;107m");
325          gotoxy(10,2);
326          printf("\033[35m\033[1m-< CADASTRAMENTO DE PACIENTE DIAGNOSTICADO COM COVID-19 >----------------<
DATA: %d - %d - %d >--\n",diaReal, mesReal ,anoReal);
327          zeraTesteOK(); // esvasia a variavel teste ok
328          testaSohCaracteres(20 ,15); // captura teclado e testa se sao soh caracteres
329          strcpy(pesquisa, testoOk); // coloca a strinq aprovada na variavel pesquisa
330          strcat(pesquisa,extencao); // concatena oq havia a em pesquisa com a extensao .txt
331          strcat(caminho,pesquisa);  // concatena oq havia a em caminho com pesquisa e forma o caminho
completo do arquivo
332          arquivo = fopen(caminho,"r"); // procura arquivo digitado, se acha abre para leitura se nao,
retorna NULL;
333          fclose(arquivo); // fecha o arquivo
334          if(arquivo!=NULL) // caso enconte o conteudo de arquivo é diferente de NULL e apaga arquivo
335          {
336             system("cls");
337             cabecalho();
338             margens();
339             gotoxy(10,2);
340             printf("\033[35m\033[1m-< CADASTRAMENTO DE PACIENTE DIAGNOSTICADO COM COVID-19
>----------------< DATA: %d - %d - %d >--\033[34m\n",diaReal, mesReal ,anoReal);
341             gotoxy(10,10);
342             remove(caminho);
343             printf("\033[31m\033[1;103m            Cadastro removido com sucesso!    \033[1;107m");
344             gotoxy(55,10);
345             system("pause");
346          }else{ // caso nao enconte o arquivo retorna Null, mostra na tela arquivo nao encontrado
347                system("cls");
348                cabecalho();
349                margens();
350                gotoxy(10,2);
351                printf("\033[35m\033[1m-< CADASTRAMENTO DE PACIENTE DIAGNOSTICADO COM COVID-19
>----------------< DATA: %d - %d - %d >--\033[34m\n",diaReal, mesReal ,anoReal);
352                gotoxy(10,10);
353                printf("\033[31m\033[1;103m           Arquivo nao encontrado!       \033[1;107m");
354               gotoxy(50,10);
355               system("pause");
356              }
357          MenuPrincipal();
358       }
359       void capturaDados(){ //Captura os dados dos arquivos criados e printa na tela
360        char extencao[10] = ".txt"; // extensao do arquivo a ser procurado
361        char caminho[500]="Pacientes/"; // pasta onte estao os arquivos dos pacientes
362        char pesquisa[500],texto[1000];
363        int w;
364        char aux0[500],aux1[500],aux2[500],aux3[500],aux4[500],aux5[500],aux6[500],aux7[500],aux8[500],
365        aux9[500],aux10[500],aux11[500],aux12[500],aux13[500],aux14[500],aux15[500],aux16[500],aux17[500];
366        FILE *arquivo;
367          consultar();
368          gotoxy(17,6);
369          printf("_____________________________________________________________");
370          gotoxy(17,7);
371          printf("|                                                            |");
372          gotoxy(17,8);
373          printf("|                                                            |");
374          gotoxy(17,9);
375          printf("|____________________________________________________________|");
376          gotoxy(29,8);
377          printf("\033[35m\033[1m-----------------< BUSCAR E LER ARQUIVO >-----------------\033[34m");
378          gotoxy(10,2);
379          printf("\033[35m\033[1m-< CADASTRAMENTO DE PACIENTE DIAGNOSTICADO COM COVID-19 >----------------<
DATA: %d - %d - %d >--\033[34m\n",diaReal, mesReal ,anoReal);
380          zeraTesteOK(); // esvasia a variavel teste ok
```

```
381          testaSohCaracteres(20 ,15); // captura teclado e testa se sao soh caracteres
382          strcpy(pesquisa, testoOk); // coloca a stringq aprovada na variavel pesquisa
383          strcat(pesquisa,extencao); // concatena oq havia a em pesquisa com a extensao .txt
384          strcat(caminho,pesquisa); // concatena oq havia a em caminho com pesquisa e forma o caminho
completo do arquivo
385          arquivo = fopen(caminho,"r"); // procura arquivo digitado, se acha abre para leitura se nao,
retorna NULL;
386          if(arquivo!=NULL) // Caso arquivo nao esta vazio
387          {
388                       for(w=0;w<18;w++) // pesquisa linha a linha do arquivo do paciente,  coloca o valor
string de cada linha, nas variaveis aux
389                       {
390                            fgets(texto,1000,arquivo);
391                            if(w==0)
392                                 strcpy(aux0,texto);
393                            if(w==1)
394                                 strcpy(aux1,texto);
395                            if(w==2)
396                                 strcpy(aux2,texto);
397                            if(w==3)
398                                 strcpy(aux3,texto);
399                            if(w==4)
400                                 strcpy(aux4,texto);
401                            if(w==5)
402                                 strcpy(aux5,texto);
403                            if(w==6)
404                                 strcpy(aux6,texto);
405                            if(w==7)
406                                 strcpy(aux7,texto);
407                            if(w==8)
408                                 strcpy(aux8,texto);
409                            if(w==9)
410                                 strcpy(aux9,texto);
411                            if(w==10)
412                                 strcpy(aux10,texto);
413                            if(w==11)
414                                 strcpy(aux11,texto);
415                            if(w==12)
416                                 strcpy(aux12,texto);
417                            if(w==13)
418                                 strcpy(aux13,texto);
419                            if(w==14)
420                                 strcpy(aux14,texto);
421                            if(w==15)
422                                 strcpy(aux15,texto);
423                            if(w==16)
424                                 strcpy(aux16,texto);
425                            if(w==17)
426                                 strcpy(aux17,texto);
427                       }
428                       // Limpa a tela escre textos dos campos e coloca os valores capturaod acima nos campos
corretamente
429                       system("cls");
430                       cabecalho();
431                       margens();
432                       gotoxy(10,2);
433                       printf("\033[35m\033[1m-< CADASTRAMENTO DE PACIENTE DIAGNOSTICADO COM COVID-19
>----------------< DATA: %d - %d - %d >--\033[34m\n",diaReal, mesReal ,anoReal);
434                       gotoxy(7,8);
435                       printf("Nome:");
436                       gotoxy(7,10);
437                       printf("Cpf:");
438                       gotoxy(7,12);
439                       printf("Data de nascimento:");
440                       gotoxy(45,12);
441                       printf("Idade:");
```

```
442                     gotoxy(56,12);
443                     printf("anos");
444                     gotoxy(55,14);
445                     printf("Email:");
446                     gotoxy(7,14);
447                     printf("Telefone:");
448                     gotoxy(7,16);
449                     printf("CEP:");
450                     gotoxy(7,18);
451                     printf("Estado:");
452                     gotoxy(55,18);
453                     printf("Cidade:");
454                     gotoxy(7,20);
455                     printf("Rua:");
456                     gotoxy(85,20);
457                     printf("Numero:");
458                     gotoxy(7,22);
459                     printf("Data do diagnostico:");
460                     gotoxy(7,24);
461                     printf("Diabetes:               Obesidade:              Hipertensao:
Tuberculose:");
462                     gotoxy(7,26);
463                     printf("Outros:");
464                     gotoxy(13,8);
465                     printf("\033[31m%s",aux0);
466                     gotoxy(12,10);
467                     printf("%s",aux1);
468                     gotoxy(27,12);
469                     printf("%c%c/%c%c/%c%c%c%c",aux2[0],aux2[1],aux2[2],aux2[3],aux2[4],aux2[5],aux2[6],
aux2[7]);
470                     gotoxy(52,12);
471                     printf("%s",aux3);
472                     gotoxy(56,12);
473                     printf("anos");
474                     gotoxy(17,14);
475                     printf("%s",aux5);
476                     gotoxy(62,14);
477                     printf("%s",aux4);
478                     gotoxy(12,16);
479                     printf("%s",aux6);
480                     gotoxy(15,18);
481                     printf("%s",aux7);
482                     gotoxy(63,18);
483                     printf("%s",aux8);
484                     gotoxy(12,20);
485                     printf("%s",aux9);
486                     gotoxy(93,20);
487                     printf("%s",aux10);
488                     gotoxy(28,22);
489                     printf("%c%c/%c%c/%c%c%c%c",aux11[0],aux11[1],aux11[2],aux11[3],aux11[4],aux11[5],
aux11[6],aux11[7]);
490                     gotoxy(18,24);
491                     printf("%s",aux12);
492                     gotoxy(45,24);
493                     printf("%s",aux13);
494                     gotoxy(71,24);
495                     printf("%s",aux14);
496                     gotoxy(101,24);
497                     printf("%s",aux15);
498                     gotoxy(15,26);
499                     printf("%s",aux16);
500                     if(aux17[0]=='1')
501                     {
502                        gotoxy(68,12);
503                        printf("\033[1;103m\033[1;32m    NAO PERTENCE AO GRUPO DE RISCO
\033[1;107m\033[31m");
```

```
504                     }
505                     else{
506                        gotoxy(68,12);
507                        printf("\033[31m\033[1;103m    PACIENTE PERTENCENTE AO GRUPO DE RISCO
\033[1m\033[31m\033[1;107m",P.idade);
508                     }
509                     gotoxy(7,28);
510                     fclose(arquivo);
511                     system("pause");
512                     MenuPrincipal();
513          }else{  // caso nao encontre o arquivo o retorno da variavel arquivo é NULL e o software printa
arquivo nao encontrado
514                 system("cls");
515                 cabecalho();
516                 margens();
517                 gotoxy(10,2);
518                 printf("\033[35m\033[1m-< CADASTRAMENTO DE PACIENTE DIAGNOSTICADO COM COVID-19
>----------------< DATA: %d - %d - %d >--\033[34m\n",diaReal, mesReal ,anoReal);
519                 gotoxy(10,10);
520                 printf("\033[31m\033[1;103m           Arquivo nao encontrado!       \033[1;107m");
521                gotoxy(50,10);
522                fclose(arquivo);
523                system("pause");
524                MenuPrincipal();
525               }
526      }
527
528      void cadastraDadosPessoais(){ // desenha a tela de cad pessoal e captura do teclado o valor dos campos
529          cabecalho();
530          linhacheia2();
                        //
531          linhaPontas2();
532          linhaPontas2();
533          printf("|              ENTER - CONFIRMA CAMPO DIGITADO           |              ESQ - APAGA
CAMPO DIGITADO                |\n");
534          linhaPontas3();
535          linhaPontas2();
536          printf("|    \033[33mCAD PESSOAL - ANDAMENTO \033[34m|    \033[31mCAD ENDERECO - N PREENCHIDO
\033[34m|    \033[31mCAD COMORBIDADES - N PREENCHIDO \033[34m|                |\n");
537          printf("|
|_____________________________________________________________________________|\n");
538          linhaPontas2();
539          linhaPontas2();
540          printf("|   Nome completo: \n");
541          linhaPontas3();
542          printf("\n|   CPF(Somente numeros 11122233344): \n");
543          linhaPontas3();
544          printf("\n|   Data de Nascimento (DDMMAAAA): \n");
545          linhaPontas3();
546          printf("\n|   Telefone para contato(51988888888): \n");
547          linhaPontas3();
548          printf("\n|   Email: \n");
549          linhaPontas3();
550          gotoxy(10,2);
551          printf("\033[35m\033[1m-< CADASTRAMENTO DE PACIENTE DIAGNOSTICADO COM COVID-19 >----------------<
DATA: %d - %d - %d >--\033[34m\n",diaReal, mesReal ,anoReal);
552          printf("\033[31m");
553          zeraTesteOK(); // zera a variavel testeok
554          testaSohCaracteres(19 ,15);       // testa para aceitar somente caractres
555          strcpy(P.nome, testoOk);       // coloca o valor de testeok - q foi aprovado no teste - em
P[i].nome
556          fflush(stdin); // zera buffer de teclado
557          zeraTesteOK(); // zera a variavel testeok
558          testaSohNumeros(38 ,18,11);            // testa para acaeitar somente numeros
559          strcpy(P.cpf, testoOk);          // coloca o valor testado em - em P.cpf
560          fflush(stdin); // zera buffer de teclado
```

```
561          zeraTesteOK(); // zera a variavel testeok
562          testaData(35 ,21); // testa para acaeitar somente numeros
563          strcpy(P.datanasc, testoOk); // coloca o valor testado em - em P.datanasc
564          fflush(stdin); // zera buffer de teclado
565          zeraTesteOK(); // zera a variavel testeok
566          testaSohNumeros(40,24,11); // testa para acaeitar somente numeros
567          strcpy(P.telefone, testoOk); // coloca o valor testado em - em P.telefone
568           fflush(stdin); // zera buffer de teclado
569          zeraTesteOK(); // zera a variavel testeok
570          testecampoVazio(11,27); // testa para acaeitar somente se o campo nao estiver vazio
571          strcpy(P.email, testoOk); // coloca o valor testado em - em P.email
572          fflush(stdin); // zera buffer de teclado
573          if(fluxo==0) // direciona o fluxo do programa, 0 para o proximo cad ou 1 para imprime cad
574          refazCad(1); // vai para o cad endereço
575          else
576          refazCad(4); // vai para o imprime cad
577      }
578      void cadastroEndereco(){ // desenha a tela de cad endereco e captura do teclado o valor dos campos
579          cabecalho();
580          linhacheia2();
581          linhaPontas2();
582          printf("|       ENTER - CONFIRMA CAMPO DIGITADO       |       ESQ - APAGA CAMPO DIGITADO
|\n");
583          linhaPontas3();
584          linhaPontas2();
585          linhaPontas2();
586          printf("|   \033[32m CAD PESSOAL - OK     \033[34m   |    \033[33mCAD ENDERECO - ANDAMENTO
\033[34m |    \033[31mCAD COMORBIDADES - N PREENCHIDO \033[34m|                  |\n");
587          printf(" _____|
|_____|\n");
588          linhaPontas2();
589          linhaPontas2();
590          printf("|   CEP(12345678): \n");
591          linhaPontas3();
592          printf("\n|   Estado:\n");
593          linhaPontas3();
594          printf("\n|   Cidade:\n");
595          linhaPontas3();
596          printf("\n|   Rua:\n");
597          linhaPontas3();
598          printf("\n|   Numero:\n");
599          linhaPontas3();
600          gotoxy(10,2);
601          printf("\033[35m\033[1m-< CADASTRAMENTO DE PACIENTE DIAGNOSTICADO COM COVID-19 >----------------<
DATA: %d - %d - %d >--\033[34m\n",diaReal, mesReal ,anoReal);
602          printf("\033[31m");
603          zeraTesteOK(); // zera a variavel testeok
604          testaSohNumeros(19 ,15,8); // testa para acaeitar somente numeros
605          strcpy(P.cep, testoOk); // coloca o valor testado em - em P.cep
606          fflush(stdin); // zera buffer de teclado
607          zeraTesteOK(); // zera a variavel testeok
608          testaSohCaracteres(12 ,18); // testa para aceitar somente caractres
609          strcpy(P.estado, testoOk); // coloca o valor de testeok - q foi aprovado no teste - em P.estado
610          fflush(stdin); // zera buffer de teclado
611          zeraTesteOK(); // zera a variavel testeok
612          testaSohCaracteres(12 ,21); // testa para aceitar somente caractres
613          strcpy(P.cidade, testoOk); // coloca o valor de testeok - q foi aprovado no teste - em P.cidade
614          fflush(stdin); // zera buffer de teclado
615          zeraTesteOK(); // zera a variavel testeok
616          testaSohCaracteres(9, 24); // testa para aceitar somente caractres
617          strcpy(P.rua, testoOk); // coloca o valor de testeok - q foi aprovado no teste - em P.rua
618          fflush(stdin); // zera buffer de teclado
619          zeraTesteOK(); // zera a variavel testeok
620          testaSohNumeros(12 , 27,0); // testa para acaeitar somente numeros
621          strcpy(P.numero, testoOk); // coloca o valor testado em - em P.numero
622          fflush(stdin); // zera buffer de teclado
```

```
623           if(fluxo==0) // direciona o fluxo do programa, 0 para o proximo cad ou 1 para imprime cad
624           refazCad(6); // vai para o cad comob
625           else
626           refazCad(2); // vai para o imprime cad
627       }
628       void cadastroComob(){ // desenha a tela de cad comob e captura do teclado o valor dos campos
629           cabecalho();
630           linhacheia2();
631           linhaPontas2();
632           printf("|       ENTER - CONFIRMA CAMPO DIGITADO       |       ESQ - APAGA CAMPO DIGITADO
|\n");
633           linhaPontas3();
634           linhaPontas2();
635           linhaPontas2();
636           printf("|    \033[32mCAD PESSOAL - OK        \033[34m|    \033[32mCAD ENDERECO - OK
\033[34m|    \033[33mCAD COMORBIDADES - ANDAMENTO  \033[34m|                    |\n");
637           printf(" ____________________________________________________________|
      |____________________|\n");
638           linhaPontas2();
639           printf("|        \033[1;90m PARA OS ITENS COM '*' ----------<  Digite 'S' para SIM e 'N' para NAO
>------------\033[34m                        \n");
640           linhaPontas3();
641           printf("\n|   DATA DO DIAGNOSTICO (DDMMAAAA):             \n");
642           linhaPontas3();
643           printf("\n|   \033[1;90m*DIABETES:\033[34m
\033[1;90m*OBESIDADE:\033[34m                                              \n");
644           linhaPontas3();
645           printf("\n|   \033[1;90m*HIPERTENSAO:\033[34m
\033[1;90m*TUBERCULOSE:\033[34m                                            \n");
646           linhaPontas3();
647           printf("\n|   OUTROS:
                        \n");
648           linhaPontas3();
649           gotoxy(10,2);
650           printf("\033[35m\033[1m-< CADASTRAMENTO DE PACIENTE DIAGNOSTICADO COM COVID-19 >----------------<
DATA: %d - %d - %d >--\033[34m\n",diaReal, mesReal ,anoReal);
651           printf("\033[31m");
652           zeraTesteOK(); // zera a testeOk
653           testeData2(36 ,17);  // testa para acaeitar somente numeros
654           strcpy(P.datadiagn, testoOk); // coloca o valor testado em - em P.datadiagn
655           fflush(stdin); // zera buffer de teclado
656           zeraTesteOK();  // zera a testeOk
657           testaSN(15 ,20); // testa para acaeitar somente "S" ou "N"
658           strcpy(P.comob1, testoOk); // coloca o valor testado em - em P.comob1
659           fflush(stdin); // zera buffer de teclado
660           zeraTesteOK(); // zera a testeOk
661           testaSN(73 ,20); // testa para acaeitar somente "S" ou "N"
662           strcpy(P.comob2, testoOk); // coloca o valor testado em - em P.comob2
663           fflush(stdin); // zera buffer de teclado
664           zeraTesteOK(); // zera a testeOk
665           testaSN(18 ,23); //// testa para acaeitar somente "S" ou "N"
666           strcpy(P.comob3, testoOk); // coloca o valor testado em - em P.comob3
667           fflush(stdin); // zera buffer de teclado
668           zeraTesteOK(); // zera a testeOk
669           testaSN(75,23); // testa para acaeitar somente "S" ou "N"
670           strcpy(P.comob4, testoOk); // coloca o valor testado em - em P.comob4
671           fflush(stdin); // zera buffer de teclado
672           zeraTesteOK(); // zera a testeOk
673           testecampoVazio(12,26); // testa para acaeitar somente campo nao vaziu
674           strcpy(P.comob5, testoOk); // coloca o valor testado em - em P.comob5
675           fflush(stdin); // zera buffer de teclado
676           refazCad(3); // vai para impreme cadastro
677       }
678       void refazCad(int c){ // Essa funcao permite refazer um cadastro, ela recebe um valor inteiro
679                             //que serve de guia de fluxo para qual tela deve ir
680           system("color e4");
```

```
681  ini2:    for(j=0;j<3;j++)
682           {
683           gotoxy(1,6);
684           printf("\033[41m
                                  ");
685           gotoxy(1,7);
686           printf("\033[41m
                                  ");
687           gotoxy(1,8);
688           printf("\033[41m
                                  ");
689           Sleep(100);
690           gotoxy(1,7);
691           printf("          \033[37m     \033[41m \033[1m Confirma formulario em - 'S' - ou Edite ele em  -
'N' -                                          ");
692           Sleep(200);
693           }
694           gotoxy(44,7);
695           tecla=getch();
696           if((tecla==115)||(tecla==83)) // le teclado e analiza o retorno, 115 e 83 - a 's' e 110 e 78 - a
'n',
697           {                             // de acordo com o valor recebido na int c, guia para a tela
requerida
698               if(c==1)
699                cadastroEndereco();
700               if(c==2)
701                imprimeCadastro();
702               if((c==3)||(c==4))
703                imprimeCadastro();
704               if(c==6)
705                  cadastroComob();
706               system("color f1");
707           }
708           else if((tecla==110)||(tecla==78))
709           {
710              if(c==1)
711                  cadastraDadosPessoais();
712              if(c==2)
713                  cadastroEndereco();
714              if(c==3)
715                  cadastroComob();
716              if(c==4)
717               cadastraDadosPessoais();
718              if(c==6)
719                cadastroEndereco();
720              system("color f1");
721           }
722           else
723              goto ini2;
724      }
725      void imprimeCadastro(){ // essa função print na tela os dados recem criados e permite 5 osoes de acao
726       fluxo=1;
727      cabecalho();
728      printf("|
                     |\n");
729      printf("|               1 - PARA SALVAR O CADASTRO         |         5 - PARA APAGAR TODO O CADASTRO
DESTA FICHA              |\n");
730      printf("|
                     |\n");
731      printf("|      2 - PARA EDITAR CAD PESSOAL    |     3 - PARA EDITAR CAD ENDERECO    |     4 - PARA
EDITAR CAD COMORBIDADES    |\n");
732      linhaPontas3();
733      printf("\033[1;94m");
734       gotoxy(3,11);
735       printf("Nome:");
736       gotoxy(3,13);
```

```
737         printf("CPF:");
738         gotoxy(25,13);
739         printf("Data de nasciemnto:");
740         gotoxy(03,15);
741         printf("Telefone:");
742         gotoxy(50,15);
743         printf("Email:");
744         gotoxy(03,17);
745         printf("CEP:");
746         gotoxy(03,19);
747         printf("Estado:");
748         gotoxy(65,19);
749         printf("Cidade:");
750         gotoxy(03,21);
751         printf("Rua:");
752         gotoxy(77,21);
753         printf("Numero:");
754         gotoxy(03,23);
755         printf("Data do diagnostico:");
756         gotoxy(03,25);
757         printf("Diabetes:                    Obesidade:                    Hipertencao:
Tuberculose:");
758         gotoxy(03,27);
759         printf("Outros:\n");
760         linhaPontas3();
761         printf("\033[31m");
762         gotoxy(9,11);
763         printf("\033[1;31m%s\033[1;34m",P.nome);
764         gotoxy(8,13);
765         printf("\033[1;31m%s\033[1;34m",P.cpf);
766         gotoxy(45,13);
767         printf("\033[1;31m%c%c/%c%c/%c%c%c%c", P.datanasc[0],P.datanasc[1],P.datanasc[2],P.datanasc[3],P.
datanasc[4],P.datanasc[5],P.datanasc[6],P.datanasc[7]);
768         gotoxy(57,13);
769         printf("\033[1;31m%d anos de idade\033[1;34m",P.idade);
770         if(idade>65){
771             gotoxy(78,13);
772             printf("\x1b[31m\033[1;103m PACIENTE PERTENCENTE AO GRUPO DE RISCO \033[1m\033[31m\033[1;107m",P.
idade);}
773          gotoxy(13,15);
774          printf("\033[1;31m%s\033[1;34m",P.telefone);
775          gotoxy(57,15);
776          printf("\033[1;31m%s\033[1;34m",P.email);
777          gotoxy(8,17);
778          printf("\033[1;31m%s\033[1;34m",P.cep);
779          gotoxy(11,19);
780          printf("\033[1;31m%s\033[1;34m",P.estado);
781          gotoxy(73,19);
782          printf("\033[1;31m%s\033[1;34m",P.cidade);
783          gotoxy(8,21);
784          printf("\033[1;31m%s\033[1;34m",P.rua);
785          gotoxy(85,21);
786          printf("\033[1;31m%s\033[1;34m",P.numero);
787          gotoxy(24,23);
788          printf("\033[1;31m%c%c/%c%c/%c%c%c%c", P.datadiagn[0],P.datadiagn[1],P.datadiagn[2],P.datadiagn[3],P
.datadiagn[4],P.datadiagn[5],P.datadiagn[6],P.datadiagn[7]);
789          gotoxy(13,25);
790          printf("\033[1;31m%s\033[1;34m",P.comob1);
791          gotoxy(43,25);
792          printf("\033[1;31m%s\033[1;34m",P.comob2);
793          gotoxy(75,25);
794          printf("\033[1;31m%s\033[1;34m",P.comob3);
795          gotoxy(107,25);
796          printf("\033[1;31m%s\033[1;34m",P.comob4);
797          gotoxy(11,27);
798          printf("\033[1;31m%s\033[1;34m",P.comob5);
```

```
799       gotoxy(10,2);
800       printf("\033[35m\033[1m-< CADASTRAMENTO DE PACIENTE DIAGNOSTICADO COM COVID-19 >----------------<
DATA: %d - %d - %d >--\033[34m\n",diaReal, mesReal ,anoReal);
801       verificaNum(16,6); // verifica qual numero e guia para a acao digitado
802       return 0;
803       }
804
805       void verificaNum(int x, int y){ // verifica qual numero e guia para a acao digitado
806
807       char extencao[10] = ".txt";
808       char caminho[500]="Pacientes/";
809       char caminho2[500]="P_Grupo_de_Risco/Pacientes do grupo de risco do dia - ";
810       char traco[10] = "-";
811       char aux[500];
812        char aux2[500];
813       FILE *arquivo;
814       FILE *arquivo2;
815       arquivo = NULL;
816       arquivo2 = NULL;
817       char data1[10];
818       char data2[10];
819       char data3[10];
820                v: tecla=0;
821                   gotoxy(x,y);
822                   tecla=getch();
823                   fflush(stdin);
824                   if(tecla==49) // digitado 1
825                   {   cabecalho();
826                       margens();
827                       gotoxy(10,2);
828                       printf("\033[35m\033[1m-< CADASTRAMENTO DE PACIENTE DIAGNOSTICADO COM COVID-19
>----------------< DATA: %d - %d - %d >--\033[34m\n",diaReal, mesReal ,anoReal);
829                           strcpy(aux,P.nome); // o arquivo é criado com o nome guardado na variavel P.nome
830                           strcat(aux,extencao);
831                           strcat(caminho,aux);
832                       arquivo = fopen(caminho,"w"); // cria arquivo e salva o os dados nele
833                       fprintf(arquivo,"%s\n",P.nome);
834                       fprintf(arquivo,"%s\n",P.cpf);
835                       fprintf(arquivo,"%s\n",P.datanasc);
836                       fprintf(arquivo,"%d\n",P.idade);
837                       fprintf(arquivo,"%s\n",P.email);
838                       fprintf(arquivo,"%s\n",P.telefone);
839                       fprintf(arquivo,"%s\n",P.cep);
840                       fprintf(arquivo,"%s\n",P.estado);
841                       fprintf(arquivo,"%s\n",P.cidade);
842                       fprintf(arquivo,"%s\n",P.rua);
843                       fprintf(arquivo,"%s\n",P.numero);
844                       fprintf(arquivo,"%s\n",P.datadiagn);
845                       fprintf(arquivo,"%s\n",P.comob1);
846                       fprintf(arquivo,"%s\n",P.comob2);
847                       fprintf(arquivo,"%s\n",P.comob3);
848                       fprintf(arquivo,"%s\n",P.comob4);
849                       fprintf(arquivo,"%s\n",P.comob5);
850                       fprintf(arquivo,"%d\n",P.risco);
851                       fclose(arquivo);
852                       gotoxy(10,10);
853                       printf("\033[31m\033[1mCadastro salvo com sucesso!");
854                       if(idade>65){ // caso idade maior q 65 sava cep e idade em arquivo separado
855                           itoa(diaReal,data1,10);
856                           itoa(mesReal,data2,10);
857                           itoa(anoReal,data3,10);
858                           strcat(data1,traco);
859                           strcat(data1,data2);
860                           strcat(data1,traco);
861                           strcat(data1,data3);
862                           strcat(caminho2,data1);
```

```
863                         strcat(caminho2,extencao);
864                         arquivo2 = fopen(caminho2,"a");
865                         fprintf(arquivo,
"|-----------------------------------------------------------------------|\n\n");
866                         fprintf(arquivo,"   Idade do paciente: %d anos\n\n",P.idade);
867                         fprintf(arquivo,"   CEP: %s \n\n",P.cep);
868                         fprintf(arquivo,
"|-----------------------------------------------------------------------|\n\n");
869                         fclose(arquivo2);
870                     }
871                     gotoxy(40,10);
872                     system("pause");
873                     MenuPrincipal();
874                 }
875                 if(tecla==50) // digitado 2 chama cad pessoal para edicao
876                     cadastraDadosPessoais();
877                 if(tecla==51) // digitado 3 chama cad endereco para edicao
878                     cadastroEndereco();
879                 if(tecla==52) // digitado 4 chama cad comob para edicao
880                     cadastroComob();
881                 if(tecla==53) // digitado 5 zera cad
882                  zeraCadastro();
883                 goto v;
884         }
885
886
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////
887
888     void zeraCadastro(){ // Zera todos os dados do digitados
889         system("color e4");
890         gotoxy(0,10);
891         printf("\033[41m
                                \n");
892         printf("\033[41m\033[37m\033[1m              TEM CERTEZA QUE VOCE QUER APAGAR TODO ESSE CADASTRO?
9 PARA SIM | \033[37mQUALQUER TECLA PARA SAIR \033[33m|           |\n");
893         printf("\033[41m
                                \n");
894         gotoxy(84,11);
895         tecla=getch();
896         fflush(stdin);
897         if(tecla==57)
898         {
899         zeraTesteOK();
900         strcpy(P.nome, testoOk);
901         strcpy(P.cpf, testoOk);
902         strcpy(P.datanasc, testoOk);
903         strcpy(P.datadiagn, testoOk);
904         strcpy(P.telefone, testoOk);
905         strcpy(P.email, testoOk);
906         strcpy(P.cep, testoOk);
907         strcpy(P.estado, testoOk);
908         strcpy(P.cidade, testoOk);
909         strcpy(P.rua, testoOk);
910         strcpy(P.numero, testoOk);
911         strcpy(P.comob1, testoOk);
912         strcpy(P.comob2, testoOk);
913         strcpy(P.comob3, testoOk);
914         strcpy(P.comob4, testoOk);
915         strcpy(P.comob5, testoOk);
916         P.idade=0;
917         P.risco=0;
918         idade=0;
919         system("cls");
920                     cabecalho();
921                     margens();
```

```
922                     gotoxy(10,2);
923                     printf("\033[35m\033[1m-< CADASTRAMENTO DE PACIENTE DIAGNOSTICADO COM COVID-19
>----------------< DATA: %d - %d - %d >--\033[34m\n",diaReal, mesReal ,anoReal);
924                     gotoxy(10,15);
925                     printf("Cadastro APAGADO com sucesso!  ");
926                     system("pause");
927                     MenuPrincipal();
928         }
929         else if(tecla!=0)
930             imprimeCadastro();
931     }
932
933     void cabecalho(){ // funcao auxiliar desenha cabeçalho
934         system("cls");
935         system("color F1");
936         linhacheia1();
937         linhaPontas1();
938         linhaPontas1();
939         linhaPontas1();
940         linhacheia1();
941     }
942
943     void margens(){ // funcao auxiliar desenha margens
944         linhacheia2();
                        //
945         linhaPontas2();
946         linhaPontas2();
947         linhaPontas2();
948         linhaPontas2();
949         linhaPontas2();
950         linhaPontas2();
951         linhaPontas2();
952         linhaPontas2();
953         linhaPontas2();
954         linhaPontas2();
955         linhaPontas2();
956         linhaPontas2();
957         linhaPontas2();
958         linhaPontas2();
959         linhaPontas2();
960         linhaPontas2();
961         linhaPontas2();
962         linhaPontas2();
963         linhaPontas2();
964         linhaPontas2();
965         linhaPontas2();
966         linhaPontas2();
967         linhaPontas3();
968     }
969
970     void linhacheia1(){ // funcao auxiliar desenha linha cheia com #
971              for(j=0;j<119;j++)     // constroe linha cheia
972             {
973                 printf("#");
974             }
975              printf("\n");
976     return 0;
977     }
978
979     void linhaPontas1(){ // funcao auxiliar desenha linha com # nas pontas e ' ' no meio
980
981             for(j=0;j<119;j++)      // constroe linha pontas
982         {
983             if((j<=7)||(j>110))
984                 printf("#");
985             else
```

```
986                 printf(" ");
987             }
988             printf("\n");
989             return 0;
990         }
991
992         void linhacheia2(){ // funcao auxiliar desenha linha cheia com '_'
993                printf(" ");
994                 for(j=0;j<117;j++)     // constroe linha cheia
995                {
996                    printf("_");
997                }
998                 printf("\n");
999         return 0;
1000        }
1001
1002        void linhaPontas2(){ // funcao auxiliar desenha linha com | nas pontas e ' ' no meio
1003
1004                for(j=0;j<119;j++)      // constroe linha pontas
1005            {
1006                if((j<1)||(j>117))
1007                    printf("|");
1008                else
1009                    printf(" ");
1010            }
1011            printf("\n");
1012            return 0;
1013        }
1014
1015        void linhaPontas3(){ // funcao auxiliar desenha linha com '|' nas pontas e ' _' no meio
1016
1017                for(j=0;j<119;j++)      // constroe linha pontas
1018            {
1019                if((j<1)||(j>117))
1020                    printf("|");
1021                else
1022                    printf("_");
1023            }
1024            printf("\n");
1025            return 0;
1026        }
1027
1028       void testaSN(int x, int y ){ // Verifica para aceitar somente caracter "S" ou "N"
1029        char testeInternoCh[500];
1030        int comprimento = 0;
1031      ini3:     gotoxy(x,y);
1032                gets(testeInternoCh);
1033                fflush(stdin);
1034                comprimento =strlen(testeInternoCh);
1035                if((comprimento==0)||(comprimento>1)){
1036                    gotoxy(x,y);
1037                    printf("\x1b[31m\033[1;103m CARACTER INVALIDO!");
1038                    tecla=0;
1039                    gotoxy(x,y);
1040                    tecla=getch();
1041                    fflush(stdin);
1042                    if(tecla!=0)
1043                    printf("\033[1m\033[31m\033[1;107m                          ");
1044                    goto ini3;
1045                 }
1046                 else if(comprimento==1){
1047                       if((testeInternoCh[0]=='S')||(testeInternoCh[0]=='s')||(testeInternoCh[0]=='n')||(
testeInternoCh[0]=='N'))
1048                       {
1049                         strcpy(testoOk, testeInternoCh);
1050                       }
```

```
1051                    else{
1052                gotoxy(x,y);
1053                printf("\x1b[31m\033[1;103m CARACTER INVALIDO!");
1054                tecla=0;
1055                gotoxy(x,y);
1056                tecla=getch();
1057                fflush(stdin);
1058                if(tecla!=0)
1059                printf("\033[1m\033[31m\033[1;107m                          ");
1060                goto ini3;
1061                    }
1062             }
1063     }
1064     void zeraTesteOK(){ // zera a variavel auxiliar TesteoK
1065          for(j=0; j < strlen(testoOk);j++){ // zera testeok antes - caso haja sujeira de codigo
1066          testoOk[j]='\0';
1067          }
1068     }
1069     void testecampoVazio(int x, int y){ // Verifica se o campo esta vazio e solicita digitacao
1070
1071      char testeInternoCh[500];
1072      int comprimento = 0;
1073    ini3:     gotoxy(x,y);
1074              gets(testeInternoCh);
1075              fflush(stdin);
1076              comprimento =strlen(testeInternoCh);
1077              if(comprimento==0){
1078                  gotoxy(x,y);
1079                  printf("\x1b[31m\033[1;103m CAMPO REQUERIDO! -   Digite qualquer tecla para continuar....."
);
1080                  tecla=0;
1081                  gotoxy(x,y);
1082                  tecla=getch();
1083                  fflush(stdin);
1084                  if(tecla!=0)
1085                  printf("\033[1m\033[31m\033[1;107m
              ");
1086                  goto ini3;
1087               }
1088              else{
1089                  strcpy(testoOk, testeInternoCh);
1090              }
1091     }
1092
1093     void testaSohCaracteres(int x, int y){ // Verifica se o campo foram digitados somente caracteres
1094          char testeInternoCh[500];
1095          int resultAx=0;
1096          int resultX=0;
1097          int comprimento;
1098          do{
1099              gotoxy(x,y);
1100              gets(testeInternoCh); // captura texto do teclado
1101              fflush(stdin); // apaga buffer de teclado
1102              comprimento =strlen(testeInternoCh); // avalia o comprimento da string
1103              if(comprimento==0){ // caso igual a zero, o campo esta vazio, solicita nova digitação
1104                  gotoxy(x,y);
1105                  printf("\x1b[31m\033[1;103m CAMPO REQUERIDO! -   Digite qualquer tecla para continuar....."
);
1106                  tecla=0;
1107                  gotoxy(x,y);
1108                  tecla=getch();
1109                  fflush(stdin);
1110                  if(tecla!=0)
1111                  printf("\033[1m\033[31m\033[1;107m
              ");
1112              }
```

```
1113             else{ // caso comprimento maior que zero avalia se sao somente caracteres
1114                 for(j=0; j < comprimento;j++){
1115                 if((isalpha(testeInternoCh[j]))||(isblank(testeInternoCh[j])))
1116                 {
1117                     resultAx=0;
1118                 }
1119                 else{ resultAx=1; // achando um caracter sai do IF
1120                       break;
1121                     }
1122             }
1123             if(resultAx>0) // solicita nova digitação
1124             {
1125                 gotoxy(x,y);
1126                 printf("\x1b[31m\033[1;103m DIGITE SOMENTE TEXTO AQUI -   Digite qualquer tecla para
continuar....");
1127                 tecla=0;
1128                 gotoxy(x,y);
1129                 tecla=getch();
1130                 fflush(stdin);
1131                 if(tecla!=0)
1132                 printf("\033[1m\033[31m\033[1;107m
             ");
1133                 j=0;
1134             }
1135             else{
1136                 strcpy(testoOk, testeInternoCh); // aceita a string digitada e coloca na var testeOK
1137                 for(j=0; j <= comprimento;j++){  // esvasia a var testeinterno
1138                    testeInternoCh[j]='\0';
1139                 }
1140                 resultX=1; // encerra o teste
1141             }
1142           }
1143         }
1144         while(resultX==0);
1145     }
1146     void testaSohNumeros(int x, int y, int c){ // Verifica se o campo foram digitados somente numeros, e
recebe o tamanho da string
1147
1148     char testeInternoCh[500];
1149         int resultAx=0;
1150         int resultX=0;
1151         int comprimento;
1152
1153   rec:  do{
1154             for(j=0; j < strlen(testeInternoCh);j++){ // zera testeInternoCh
1155             testeInternoCh[j]='\0';}
1156             gotoxy(x,y);
1157             gets(testeInternoCh); // captura texto do teclado
1158             fflush(stdin); // apaga buffer de teclado
1159             comprimento =strlen(testeInternoCh); // avalia o comprimento da string
1160             if(comprimento==0){ // caso igual a zero, campo vazio,  solicita nova digitação
1161                 gotoxy(x,y);
1162                 printf("\x1b[31m\033[1;103m REQUERIDO ");
1163                 tecla=0;
1164                 gotoxy(x,y);
1165                 tecla=getch();
1166                 fflush(stdin);
1167                 if(tecla!=0)
1168                 printf("\033[1m\033[31m\033[1;107m              ");
1169             }
1170             else{
1171                 for(j=0; j < comprimento;j++){
1172                 if(isalpha(testeInternoCh[j]))
1173                 {
1174                     resultAx=1;
1175                     j=comprimento;
```

```
1176                }
1177                else{ resultAx=0;}
1178            }
1179            if(resultAx>0)
1180            {
1181                gotoxy(x,y);
1182                printf("\x1b[31m\033[1;103m SOMENTE NUMEROS ");
1183                tecla=0;
1184                gotoxy(x,y);
1185                tecla=getch();
1186                fflush(stdin);
1187                if(tecla!=0)
1188                printf("\033[1m\033[31m\033[1;107m                 ");
1189                j=0;
1190            }
1191            else{
1192                if(c==0) // avalia se C for igual a zero independe o tamanho do ada, aceita qualquer
tamanho
1193                {
1194                  strcpy(testoOk, testeInternoCh); // aceita a string digitada e coloca na var testeOK
1195                }
1196                else { if(c==(strlen(testeInternoCh))) // compara se o tado digitado tem o tamanho
requerido
1197                        {
1198                            strcpy(testoOk, testeInternoCh); // aceita a string digitada e coloca na var
testeOK
1199                            for(j=0; j <= comprimento;j++){  // zera teste interno
1200                            testeInternoCh[j]='\0';
1201                            }
1202                        }
1203                        else{ // caso o tamanho do dado seja diferente do requerido, solicita nova
digitação
1204                                gotoxy(x,y);
1205                                printf("\x1b[31m\033[1;103m NUMERO INVALIDO ");
1206                                tecla=0;
1207                                gotoxy(x,y);
1208                                tecla=getch();
1209                                fflush(stdin);
1210                                if(tecla!=0)
1211                                printf("\033[1m\033[31m\033[1;107m                              ");
1212                                goto rec;
1213                               }
1214                }
1215                                resultX=1;
1216            }
1217          }
1218        }
1219        while(resultX==0);
1220    }
1221
1222
1223    void testeData2(int x, int y){ // verifica se a data esta corretamente digitada
1224
1225    int tamData;
1226    char dia[2];
1227    char mes[3];
1228    char ano[5];
1229    int diaN = 0;
1230    int mesN = 0;
1231    int anoN = 0;
1232
1233  iniXX:    for(j=0; j < strlen(testoOk);j++){ // zera testeok antes - caso haja sujeira de codigo
1234            testoOk[j]='\0';
1235            }
1236            for(j=0; j < strlen(dia);j++){ // zera testeok antes - caso haja sujeira de codigo
1237            dia[j]='\0';
```

```
1238            }
1239            for(j=0; j < strlen(mes);j++){ // zera testeok antes - caso haja sujeira de codigo
1240            mes[j]='\0';
1241            }
1242            for(j=0; j < strlen(ano);j++){ // zera testeok antes - caso haja sujeira de codigo
1243            ano[j]='\0';
1244            }
1245            diaN=0;
1246            mesN=0;
1247            anoN=0;
1248
1249            for(j=0; j < strlen(testoOk);j++){ // zera testeok antes - caso haja sujeira de codigo
1250            testoOk[j]='\0';
1251            }
1252            testaSohNumeros(x,y,8); // testa se sao somente numeros e o comprimentoda string deve ser de 8
1253            tamData = strlen(testoOk);
1254
1255            if(tamData!=8)  // se tamanho da string for diferente de 8, solicita nova digitação
1256            {
1257                gotoxy(x,y);
1258                printf("\x1b[31m\033[1;103m DATA INVALIDA DDMMAAAA! -   Digite qualquer tecla para
continuar....");
1259                tecla=0;
1260                gotoxy(x,y);
1261                tecla=getch();
1262                fflush(stdin);
1263                if(tecla!=0)
1264                printf("\033[1m\033[31m\033[1;107m
                  ");
1265                goto iniXX;
1266            }
1267            else{ // se tamanho da string for igual a 8 e forem soh numeros aceita o dado
1268                dia[0] = testoOk[0];
1269                dia[1] = testoOk[1];
1270                mes[0] = testoOk[2];
1271                mes[1] = testoOk[3];
1272                ano[0] = testoOk[4];
1273                ano[1] = testoOk[5];
1274                ano[2] = testoOk[6];
1275                ano[3] = testoOk[7];
1276                anoN = atoi(ano);
1277                mesN = atoi(mes);
1278                diaN = atoi(dia);
1279                gotoxy(x,y);
1280                printf("%c%c / %c%c / %c%c%c%c", testoOk[0],testoOk[1],testoOk[2],testoOk[3],testoOk[4],
testoOk[5],testoOk[6],testoOk[7]); // printa a data
1281                if((diaN>31)||(mesN>12)||(anoN>2021)||(anoN<1900)) // verifica de os dados de data estao
corretos, caso contrario solicita nova digitação
1282                {
1283                gotoxy(x,y);
1284                printf("\x1b[31m\033[1;103m DATA INVALIDA DDMMAAAA! -                          ");
1285                tecla=0;
1286                gotoxy(x,y);
1287                tecla=getch();
1288                fflush(stdin);
1289                if(tecla!=0)
1290                printf("\033[1m\033[31m\033[1;107m
                ");
1291                goto iniXX;
1292                }
1293      }}
1294
1295      void testaData(int x, int y){
1296      int tamData;
1297      char dia[10];
1298      char mes[20];
```

```
1299      char ano[20];
1300      int diaN = 0;
1301      int mesN = 0;
1302      int anoN = 0;
1303      char idadeCH[50];
1304      char idadeFormatada[50];
1305        ini:  for(j=0; j < strlen(testoOk);j++){ // zera testeok antes - caso haja sujeira de codigo
1306              testoOk[j]='\0';
1307              }
1308              for(j=0; j < strlen(dia);j++){ // zera testeok antes - caso haja sujeira de codigo
1309              dia[j]='\0';
1310              }
1311              for(j=0; j < strlen(mes);j++){ // zera testeok antes - caso haja sujeira de codigo
1312              mes[j]='\0';
1313              }
1314              for(j=0; j < strlen(ano);j++){ // zera testeok antes - caso haja sujeira de codigo
1315              ano[j]='\0';
1316              }
1317              diaN=0;
1318              mesN=0;
1319              anoN=0;
1320              testaSohNumeros(x,y,8); // testa se sao somente numeros e o comprimentoda string deve ser de 8
1321              tamData = strlen(testoOk);
1322              if(tamData!=8) // se tamanho da string for diferente de 8, solicita nova digitação
1323              {
1324                  gotoxy(x,y);
1325                  printf("\x1b[31m\033[1;103m DATA INVALIDA DDMMAAAA! -   Digite qualquer tecla para
continuar....");
1326                  tecla=0;
1327                  gotoxy(x,y);
1328                  tecla=getch();
1329                  fflush(stdin);
1330                  if(tecla!=0)
1331                  printf("\033[1m\033[31m\033[1;107m
                    ");
1332                  goto ini;
1333              }
1334              else{ // se tamanho da string for igual a 8 e forem soh numeros aceita o dado
1335                  gotoxy(x,y);
1336                  printf("%c%c / %c%c / %c%c%c%c", testoOk[0],testoOk[1],testoOk[2],testoOk[3],testoOk[4],
testoOk[5],testoOk[6],testoOk[7]);// printa a data
1337
1338                  gotoxy(x,21);
1339                  diaN=0;
1340                  mesN=0;
1341                  anoN=0;
1342                  dia[0] = testoOk[0];
1343                  dia[1] = testoOk[1];
1344                  diaN = atoi(dia);
1345                  mes[0] = testoOk[2];
1346                  mes[1] = testoOk[3];
1347                  mesN = atoi(mes);
1348                  ano[0] = testoOk[4];
1349                  ano[1] = testoOk[5];
1350                  ano[2] = testoOk[6];
1351                  ano[3] = testoOk[7];
1352                  anoN = atoi(ano);
1353                  if((diaN>31)||(mesN>12)||(anoN>2021)||(anoN<1900)) // verifica de os dados de data estao
corretos, caso contrario solicita nova digitação
1354                  {
1355                  gotoxy(x,y);
1356                  printf("\x1b[31m\033[1;103m DATA INVALIDA DDMMAAAA! -   Digite qualquer tecla para
continuar....");
1357                  tecla=0;
1358                  gotoxy(x,y);
1359                  tecla=getch();
```

```
1360                fflush(stdin);
1361                if(tecla!=0)
1362                printf("\033[1m\033[31m\033[1;107m
                "");
1363                goto ini;
1364                }
1365               anoN = anoReal - anoN;  // começa calcular data de aniversario
1366               mesN = mesReal - mesN;
1367               diaN = diaReal - diaN;
1368               if(mesN<0)
1369                idade = anoN-1;
1370               if(mesN>0)
1371                 idade = anoN;
1372               if(mesN==0)
1373               {
1374                   if(diaN>=0)
1375                    idade = anoN;
1376                   else
1377                    idade = anoN-1;
1378               }                        // termina calcular Data de nascimento
1379                gotoxy(68,21);
1380                printf("Idade: %d anos ",idade);// imprime o resultado
1381                P.idade = idade;
1382
1383                if(idade>65) // verifica se idade maior que 65,
1384                    P.risco=2; //  se sim - Pasiente de risco
1385                   else
1386                    P.risco=1; // se nao - Nao é Pasiente de risco
1387           }
1388     }
1389
```