**UNIVERSIDADE PAULISTA – UNIP EaD**

**Projeto Integrado Multidisciplinar**

**Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas**

**Sistema de cadastro de pacientes com Covid-19 em C**

**Polo Campinas – Centro**

**2021**

**UNIVERSIDADE PAULISTA – UNIP EaD**

**Projeto Integrado Multidisciplinar**

**Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas**

**Sistema de cadastro de pacientes com Covid-19 em C**

**Nome:** Giovanne Piçuti Pereira da Silva

**RA:** 2186105

**Curso:** Análise e Desenvolvimento de Sistemas

**Semestre:** 2º Semestre

**Polo Campinas – Centro**

**2021**

**RESUMO**

Este projeto tem como finalidade unir as matérias linguagem e técnicas de programação e engenharia de software I para a criação de um sistema de cadastro de pacientes utilizando a linguagem de programação “C”. Com a pandemia que atingiu o mundo, muitos centros de saúde necessitam de maneiras para agilizarem o cadastramento dos pacientes que passam por lá, e muitas vezes os pacientes que fazem parte do grupo de risco acabam sendo esquecidos. A ideia do sistema é identificar se o paciente faz parte do grupo de risco e então criar um arquivo separado, armazenando nome, idade, CEP e quais comorbidades o mesmo possui.

Palavras-chave: C, cadastro, sistema, COVID-19

**ABSTRACT**

This project aims to unite the language and techniques of programming language and software engineering I for the creation of a patient registration system using the “C” programming language. With the pandemic that hit the world, many health centers need ways to streamline the registration of patients who pass through there, and often patients who are part of the risk group end up being forgotten. The system's idea is to identify if the patient is part of the risk group and then create a separate file, storing name, age, zip code and what comorbidities he has.

Keyword: C, registration, system, COVID-19

**Sumário**

[**1.** **Introdução** 6](#_Toc88995468)

[**2.** **Bibliotecas** 7](#_Toc88995469)

[2.1 STDIO.H 7](#_Toc88995470)

[2.2 STDLIB.H 7](#_Toc88995471)

[2.3 STRING.H 7](#_Toc88995472)

[2.4 LOCALE.H 8](#_Toc88995474)

[2.5 TIME.H 8](#_Toc88995475)

[**3.** **Variáveis** 9](#_Toc88995476)

[3.1 Variáveis declaradas e #define 9](#_Toc88995477)

[3.2 Variáveis time.h 10](#_Toc88995479)

[**4.** **Estrutura do código** 11](#_Toc88995480)

[4.1 Login 11](#_Toc88995481)

[4.2 Cadastro do paciente 12](#_Toc88995485)

[4.3 Data de nascimento 13](#_Toc88995486)

[4.4 Cadastro de endereço 13](#_Toc88995487)

[**5** **Gravando dados no arquivo** 14](#_Toc88995488)

[5.1 Cadastro grupo de risco 15](#_Toc88995489)

[**6.** **Conclusão** 16](#_Toc88995496)

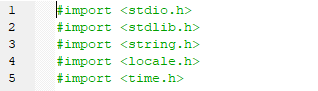
[**REFERÊNCIAS** 17](#_Toc88995497)

# **Introdução**

A pandemia global estimulou uma grande demanda pela inovação médica, o que fez diversas empresas a começarem a investir no desenvolvimento de softwares de saúde. Como o coronavírus infecta um número exponencial de pessoas, é necessário fazer um rastreamento, identificando os pacientes e principalmente aqueles que estão no grupo de risco. Pacientes de grupo de risco precisam de um cuidado maior que outros pacientes, tendo em vista que o coronavírus se mostra muito mais letal com eles.

Esse sistema foi pensado justamente no cadastramento de pacientes, onde é coletado os dados dos mesmos e cria um arquivo em formato “.txt”.  
O sistema também identifica quando o paciente faz parte do grupo de risco. Se possuir a idade igual ou superior a 65 anos e/ou possuir comorbidades, ele gera um outro arquivo em .txt contendo o nome, idade, CEP e quais comorbidades esse paciente possui, e então esse arquivo pode ser enviado para central de Secretaria da Saúde da cidade.

# **Bibliotecas**



**Figura 1:** Bibliotecas. Fonte: Autor próprio.

No começo do programa é onde iremos dar #import ou #include nas principais bibliotecas que iremos utilizar. As bibliotecas são responsáveis para a utilização de diversos comandos diferentes dentro do programa, não só em C mas em outras linguagens também.

## STDIO.H

A biblioteca <stdio.h> é a principal biblioteca para utilização do C, ela é responsável para tratamento de entrada e saída de dados. A saída de um dado é o que é mostrado na tela quando é digitado no programa utilizando por exemplo o comando printf (que é o print de imprimir/mostrar em inglês, e o F de formatado.)

Quando falamos de entrada de dados estamos falando do que o usuário digita no programa, podemos utilizar o scanf( ) e gets( ) por exemplo.

Sem a biblioteca stdio.h não seria possível estar utilizando esses comandos e nem tratando esses dados.

## STDLIB.H

A biblioteca stdlib.h é responsável pela implementação de diversas funções para diversas operações, como conversão, alocação de memória, controle de processo, funções de busca e ordenação. Utilizamos a stdlib.h para utilizar o comando “system”.

## STRING.H

A biblioteca string.h serve para o tratamento de string. Podemos utilizar diversos comandos como copiar uma string, concatenar, descobrir o tamanho de uma string, ou, como foi utilizado no código, comparar duas strings diferentes.



## LOCALE.H

A biblioteca locale.h tem como principal função constantes especificas de acordo com uma localização especifica. Como em nosso idioma utilizamos muito acentuações e caracteres especiais como “Ç”, é necessário fazer a importação dessa biblioteca.

## TIME.H

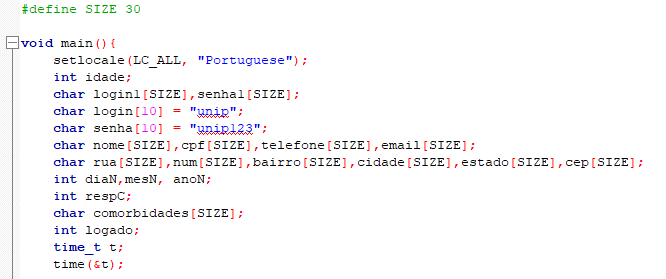
A biblioteca time.h tem função tratamento de data e hora. Utilizamos ela no programa para podermos pegar a data que o programa está sendo utilizado para dar a data do diagnóstico e também já fazer a conta para identificar a idade do paciente e conferir se o mesmo está dentro do grupo de risco ou não.

# **Variáveis**

Variáveis são espaços na memória do computador, dedicados a um dado que serão alterados durante a execução do mesmo. Variáveis precisam ser definidas por nomes e tipos e seguindo algumas regras para a criação da mesma. Para dar nome a uma variável sempre deve começar com letras, podem possuir números inteiros, único símbolo que pode se possuir é o “\_”, não pode contar espaço, não podem conter acentos e não podem ser palavras restritas.

Variáveis podem ser declaradas como números inteiros (0, 1, 2, 3...), números reais, que podem ser números inteiros e decimais (0,1; 5,2; 3), também podem ser de tipo caracteres, que podem ser número reais, letras e outros símbolos e também podem ser do tipo lógico, como VERDADEIRO ou FALSO.

## Variáveis declaradas e #define



**Figura 2:** Variáveis declaradas e #define. Fonte: Autor próprio.

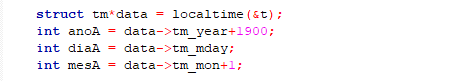
Antes de declarar as variáveis, foi criado uma diretiva no começo do programa chamada SIZE, com tamanho 30. Uma diretiva é um macro, uma pequena associação de um identificador ou parâmetro criado no começo do programa, esse SIZE foi utilizado apenas para utilizar nas variáveis para definir o tamanho delas.

Logo no começo, utilizamos o setlocale (LC\_ALL, “Portuguese”). apesar de não ser uma variável, como foi feito a importação da biblioteca locale.h, é necessário fazer a diretriz dizendo qual é o parâmetro que vamos seguir. O LC\_ALL faz referência a todos os aspectos da localização, e “Portuguese” mostrando que estamos importando os aspectos da língua portuguesa.

Após isso, fizemos a declaração das principais variáveis, a idade como tipo inteiro, onde o valor que será atribuído será adquirido após a conta do ano atual juntamente com o ano de nascimento do paciente. Login1 e Senha1 foram as variáveis que serão utilizadas para leitura de dados e como será para o login utilizamos o tipo char. Logo em seguida colocamos “Char login[10] = unip” e “char senha = unip123”, isso foi feito definindo já a senha e login que será usado no usuário. Os principais dados dos usuários como nome, CPF, telefone, e-mail e endereço foi colocado como char e usando a definitiva SIZE para colocar o tamanho padrão em todas elas. Na parte do nascimento, fizemos a utilização do tipo int, para declarar que são números inteiros que serão digitados.



## Variáveis time.h



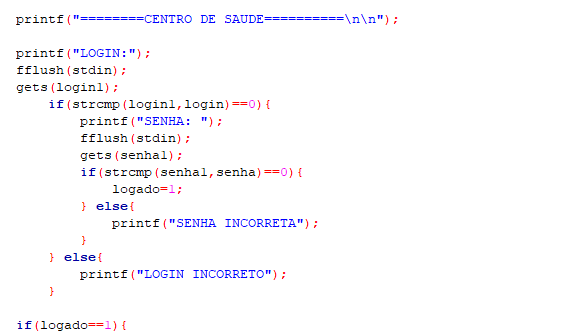
**Figura 3:** Variáveis time.h. Fonte: Autor próprio.

Aqui foram feitos os primeiros ajustes para utilização da biblioteca time.h. Como o programa irá mostrar o dia que foi feito o diagnóstico do paciente, precisamos criar uma struct, com um ponteiro para armazenar a data. Utilizamos então o tipo inteiro para anoA (ano atual), como C utiliza o ano 0 o ano de 1900, é necessário declarar tm\_year (para pegar o ano)+1900, assim obtemos o ano atual. Na parte do mês, o C conta janeiro como mês 0, e não mês 1, então precisamos fazer tm\_mon (para obter o mês)+1. Na parte do dia não há problema, então pode utilizar o tm\_mday para obter o dia do mês, indo de 1 a 31.

# **Estrutura do código**

Feito a importação das bibliotecas e declaração das variáveis, começamos a desenvolver e pensar de que forma o código irá atuar e funcionar, onde e como iremos utilizar as variáveis e quais estruturas de condições iremos usar.

## Login



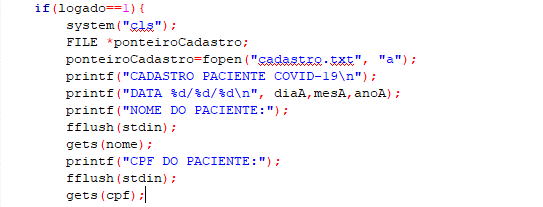
**Figura 4:** Login. Fonte: Autor próprio.

No começo do código, ao executar o .exe, irá aparecer no começo do programa pedindo o login e a senha do usuário. Como foi feito a declaração da variável contendo os valores que precisam ser digitados, utilizamos a estrutura If. O If é uma estrutura de condição que verifica se o que está sendo passado como diretriz é verdadeiro. O usuário digita o login, usamos o comando strcmp (login1,login), esse comando é feito graças a biblioteca string.h, que compara se duas strings são iguais, caso sejam iguais ela retorna o valor 0, então, o IF faz a verificação que, se o valor retornado pelo strcmp for 0, ele irá dar continuidade e irá pedir a senha do usuário. Caso o valor digitado seja diferente de 0, irá aparecer a mensagem “LOGIN INCORRETO” e o programa irá fechar.

Na parte da senha, utilizamos também outro If(strcmp), mas nesse, caso a senha digitada desse 0, iria adicionar o valor 1 para a variável logado, que foi declarada logo no começo. Essa variável é a que irá fazer o código dar continuidade.



## Cadastro do paciente



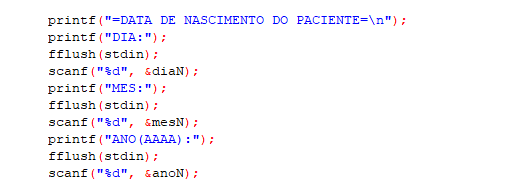
**Figura 5:** Cadastro do paciente. Fonte: Autor próprio.

Assim que o profissional de saúde fez o login, e o login e a senha estavam iguais, o If da senha irá adicionar a variável “logado” o valor 1. O código ao identificar que “logado” recebeu o valor 1, irá abrir um outro IF, onde será dado o início do cadastro do paciente.

Primeiro foi colocado um system(“cls”), esse comando foi usado para limpar a tela e tirar as informações do login, logo em seguida foi criado um ponteiro do tipo FILE, para que todos os dados do paciente possam estar sendo gravados. Após a criação do ponteiro utilizamos o comando ponteiroCadastro (nosso ponteiro que criamos)=fopen(“cadastro.txt”, “a”). Primeiro, indicamentos qual ponteiros iremos usar, no nosso caso o ponteiroCadastro, em seguida usamos o comando fopen dizendo que vamos abrir, então, dentro dos parênteses indicamos qual o nome do arquivo que será criado(cadastro.txt) e em seguida indicamos a letra “a”, esse “a” é a permissão para abrir o arquivo de texto para gravação, permitindo também adicionar novos dados quando for aberto de novo o arquivo. Caso esse arquivo não exista, ele irá ser criado.

Após isso, começamos a anotar os dados básicos do paciente. Usamos o comando printf para informar na tela qual informação está sendo solicitada, o comando fflush(stdin) para limpar os buffers que foram escritos e assim escrever novos dados sem ter algum problema. Usamos também o comando gets(), pois como nas variáveis declaramos elas como char, o gets permite conseguir strings de forma mais formatada.

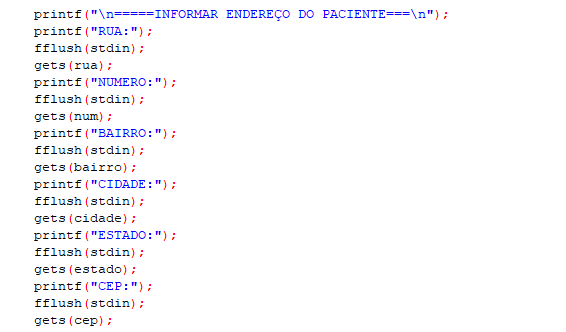
## Data de nascimento



**Figura 6:** Data de nascimento. Fonte: Autor próprio.

Para a conseguir a data de nascimento do paciente, utilizamos o formato dela dividido entre dia, mês e ano. Como declaramos a data no formato int, utilizamos o scanf, logo em seguida o símbolo de % juntamente dizendo qual o tipo de variável, o %d diz que essa variável é do tipo int, depois o & para indicar para em qual variável será gravado o dado.

## 4.4 Cadastro de endereço

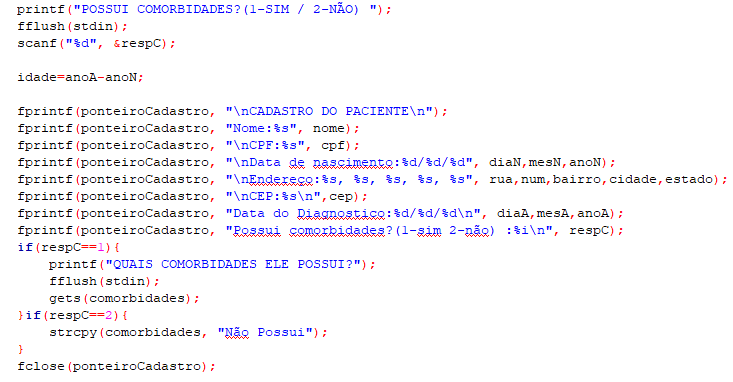


**Figura 7:** Cadastro de endereço. Fonte: Autor próprio.

Para o cadastro do endereço colocamos todas as variáveis como string para ficar mais fácil o cadastramento das variáveis. Novamente utilizamos o gets e informando em qual variável será gravado o dado digitado.

# **Gravando dados no arquivo**

Após o profissional de saúde digitar todos os dados básicos do paciente, está na hora de gravarmos ele no arquivo .txt que criamos.



**Figura 8:** Gravação dos dados no arquivo. Fonte: Autor próprio.

Antes de gravar os dados, foi colocado a pergunta se o paciente possuía comorbidades, caso o mesmo possuísse, iria ser colocado em uma pergunta em formato inteiro, para usarmos em um if.

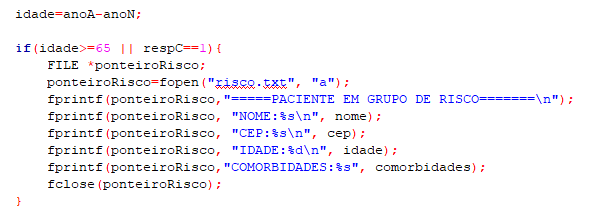
Para poder gravar os dados nos arquivos, usamos o comando fprintf, informando em qual ponteiro iremos gravar e qual a informação que iríamos usar. Novamente é utilizado o % para indicar qual o formato da variável que será gravada, se será inteira(%d) ou string(%s) e em seguida, o nome da variavel.

Fizemos também a data do diagnostico que foi obtido através da biblioteca “time.h”, utilizando o “%d” e fazendo a chamada das variáveis, que já tinham sido adquiridas no struct.

Caso a resposta sobre comorbidades fosse 1 para sim, então iria abrir um If para poder informar quais comorbidades são. Caso fosse 2 para não, usamos um IF e um outro comando da biblioteca string.h que é o “strcpy”, que copia uma palavra para uma variável, então colocamos “Não possui”.

Após ter sido gravados todos os dados no arquivo .txt de cadastro de paciente, usamos o comando fclose e o nome do ponteiro para estar fechando o arquivo.

## Cadastro grupo de risco



**Figura 9:** Cadastro de grupo de risco. Fonte: Autor próprio.

Após ser feito todo o cadastro do paciente, abrimos a variável “idade” e atribuímos a ela o valor do ano atual menos o ano de nascimento, para assim conseguirmos a idade do paciente. Abrimos um novo “if”, e nele colocamos um operador lógico OR(OU), que, caso a idade do paciente seja igual ou acima de 65 anos e/ou na pergunta se o mesmo possuía comorbidade a resposta tenha sido 1 sim, então é feito um novo ponteiro para grupo de risco, nele é adicionado o nome do paciente, CEP, e informando também a idade que ele possui. Além disso, também é colocado nesse arquivo quais comorbidades ele tem.



# **Conclusão**

Este trabalho trouxe a possibilidade de construir o primeiro sistema, ao analisar o caso, foi feito toda a estrutura dele em uma folha de papel, utilizando um diagrama, utilizando a lógica de programação podemos analisar qual a ordem que teria que ser realizada, onde que seria viável colocar os “if” e como poderíamos estar melhorando.

Foram dias de digitação, testes, falhas, mas ao final, ver um programa totalmente escrito por você funcionando traz uma grande satisfação.

# **REFERÊNCIAS**

CASAVELLA, Eduardo. **A biblioteca padrão da linguagem C** Disponível em: <http://linguagemc.com.br/a-biblioteca-padrao-da-linguagem-c/>. Acesso dia 12 de Novembro de 2021.

CASAVELLA, Eduardo. **A biblioteca string.h**. Disponível em: <http://linguagemc.com.br/a-biblioteca-string-h/>. Acesso dia 12 de novembro de 2021.

CASAVELLA, Eduardo. **Arquivos em C** Disponível em: <http://linguagemc.com.br/arquivos-em-c-categoria-usando-arquivos/>. Acesso dia 14 de novembro de 2021.

CASAVELLA, Eduardo. **Exibindo data e hora com time.h e localtime.** Disponível em: <http://linguagemc.com.br/exibindo-data-e-hora-com-time-h/>. Acesso dia 12 de novembro de 2021.

GUABABARA, Gustavo. **M03A01 – Condições Simples.** Disponível em <https://www.estudonauta.com/licao/m03a01-condicoes-simples/>. Acesso dia 10 de novembro de 2021.

GUANABARA, Gustavo. **Introdução a Algoritmos** Disponível em: <https://www.cursoemvideo.com/curso/curso-de-algoritmo/aulas/algoritmo/modulos/introducao-a-algoritmos/>. Acesso dia 01 de novembro de 2021.

PRAKASH, Pankaj. **C program to create a file and write data into file**. Disponível em: <https://codeforwin.org/2018/01/c-program-create-file-write-contents.html?fbclid=IwAR27bAsANTQk4p0Q7jrs1ve5lHTZeQuj-f7Hy3QE5IrFjYahTG5HhGmvBbQ>. Acesso dia 14 de novembro de 2021.