



UNIVERSIDADE JEAN PIAGET DE ANGOLA
FACULDADE DE CIÊNCIAS E
TECNOLOGIAS

RELATÓRIO DE PROGRAMAÇÃO I

**PROGRAMA QUE PERMITE REGISTRAR INFORMAÇÕES DE PESSOAS
PARA A VACINA CONTRA OS VIRUS DA COVID-19.**

GRUPO A1

INTEGRANTES

Lucrécio Daniel Barnabé

Luzolo Marcia da Costa Diango

Docente

Lúcia Távira P. Bravo

Luanda, 2022

ÍNDICE

INTRODUÇÃO	1
OBJETIVOS	1
Geral	1
Específico	1
1. RELATÓRIO DO TRABALHO	2
2. CÓDIGO-FONTE	5
3. CONCLUSÃO	12
4. BIBLIOGRAFIA.....	13

INTRODUÇÃO

Em virtude da pandemia provocada pelo vírus da COVID-19, é obrigatório que as pessoas estejam vacinadas. Para o efeito, devem fazer o registo para a vacinação. Após o registo é emitido um registo individual para cada pessoa registada.

O programa a seguir será capaz de efectuar o registo para vacinação de 5 pessoas no máximo, e mostrar o registo individual por meio de várias funções.

PROBLEMÁTICA

O que devemos fazer para se ter o control da população que irá tomar a vacina contra o vírus da COVID-19?

HIPÓTESE

- Para a população tomar a vacina, é obrigatório em primeiro lugar fazer o registo para a vacinação.
- Portanto desenvolveu-se um programa que permite realizar o registo da vacinação, e mostrando o registo individual.

OBJETIVOS

- Geral
 - Criar um programa em linguagem c que possibilita efetuar o registo da vacina contra a COVID-19.
- Específico
 - Fazer o registo geral;
 - Preencher a estrutura do registo individual;
 - Mostrar o Registo individual.

1. RELATÓRIO DO TRABALHO

Começamos por definir quatro bibliotecas, a primeira que foi a **stdio.h** que é uma biblioteca com funções de entrada /saída de dados, a segunda biblioteca é a **stdlib.h** para utilizar as funções do windows, a terceira biblioteca é a **string.h** serve para fazer manipulação de strings (conjuntos de caracteres), e a última biblioteca é a **time.h** que serve para manipular o tempo de processamento da CPU. Utilizamos o **define** para definir o número máximo de pessoas que podem ser registada no sistema a cada execução.

Estrutura (**struct**) é uma coleção de uma ou mais variáveis, possivelmente de tipos diferentes, que podem ser manipuladas em conjunto ou em separado. Portanto declaramos quatro estruturas com seus respectivos campos, para armazenar os dados requisitado pelo programa para armazenar os dados dos candidatos ao registo da vacina.

Criamos dezanove funções para a resolução do problema proposto, a seguir explicaremos o objectivo de cada uma delas:

- **Função Interface:** foi criada para dar estilo ao programa
- **Função pausar_ex:** serve para criar um compasso de espera de tempo entre o processamento de cada informação.
- **Função inf_etapas:** foi criada para mostrar em qual etapa o utilizador se encontra ao efectuar o registo.
- **Funções sms_1_analise:** serve para analisar se os dados foram atribuídos corretos.
- **Funções sms_2_sucesso:** serve para emitir uma sms ao utilizador quando os dados foram bem sucedidos.
- **Função sms_3_alert:** quando o utilizador estiver no menu que lhe possibilita ver os registos por partes, se ele escolher uma opção que não faz parte do menu de opções, esta função será executada alertando assim o utilizador de que os dados seleccionados não existe.
- **Função sms_4_erro:** foi criada para enviar uma mensagem de erro, quando o utilizador digitar um número superior a 5 que é o limite máximo de pessoas que o programa pode registar, ou se o utilizador digitar um número superior a 3 ou inferior a 0, no caso da comorbilidade onde o limite é 3, passando este valor esta função é executada mostrando assim ao utilizador o intervalo de número que ele deve digitar.

No entanto, antes de realizar o registo geral, o grupo A1 pensou em dividir o registo geral em quatro modulos ou etapas, criando assim quatro funções distintas, que seram explicadas a seguir.

- **Função mod_1_reg:** Ao executar esta função, primeiramente é executada a função **inf_etapas**, que irá informar ao utilizador em que etapa se encontra, posteriormente o utilizador irá realizar o cadastro de alguns dados, tais como: **Nome, documento individual, número do documento individual** e o **email**. Depois disto é executada a função **sms_2_sucesso**, para informar ao utilizador que os dados foram cadastrados com sucesso.
- **Função mod_2_reg:** diferente do mod_reg, apesar de que possui algumas funções iguais como executar as funções **inf_etapas**, e **sms_2_sucesso**. Esta função permite cadastrar, o **número de telefone e a data de nascimento**. E de seguida é executada a função **sms_1_analise** e a data estiver cumprindo a condição então é executada a função **sms_2_sucesso**, caso não imprime uma sms dizendo que a data digitada não esta correta, e assim o utilizador irá digitar novamente.
- **Função mod_3_reg:** Ao executar esta função, primeiramente é executada a função **inf_etapas**, este modulo serve para cadastrar **a nacionalidade, província e municipio**.
- **Função mod_4_reg:** Ao executar esta função, primeiramente é executada a função **inf_etapas**, neste modulo, é apresentado um menu com duas opções, o programa ira pedir ao utilizador para escolher a opção um (1) se ele tiver comorbilidade, só assim será possível, solicitando a quantidade de doença que o utilizador tem posteriormente é executado a função **sms_2_analise**, e se a quantidade satisfazer a condição, o utilizador irá digitar as comorbilidades. E por fim é executado a função **sms_2_sucesso**. Caso o uiltizador escolha a opção zero (0) o programa irá considerar que este utilizador não possui comorbilidade.
- **Função cad_vacina:** tem a responsabilidade de executar os quatros modulos, tais como: **mod_1_reg, mod_2_reg, mod_3_reg** e o **mod_4_reg**. A cada fim de um registo é emitido uma sms informando ao utilizador que o registo x foi efetuado com sucesso.

Depois de efetuar o registo geral, o programa tem a obrigatoriedade de realizar ou de preencher a estrutura do registo individual por isso criou-se a seguinte função:

- **Função preencher_ri:** Para preencher o ri foi utilizada a função strcpy que tem a função de copiar um valor contido em uma variável para outra variável. E tem a seguinte sintaxe: `strcpy(destino, origem);`
- **Mostrar_ri:** Tem a finalidade de mostrar todos os registos individuais cadastrados ao utilizador, tais como: **nome, código de registo, documento de identidade** e o **numero do documento de identidade**.
- **Mostrar_ri_por_parte:** esta função permite mostrar ao utilizador um registo de cada vez.
- **Menu_principal:** Mostra as principais funções que o utilizador tem de escolher, temos as seguintes funções: efetuar registo, opções para mostrar o registo individual e sair ou terminar a execução.
- **Op_mostrar:** Mostra um menu de opções que permite ao utilizador escolher o numero de registo que ele quer ver, ou se ele deseja ver todos de uma só vez também é possível, esta função foi declarada depois da função main, antes porém, foi criada um protótipo desta mesma função comprindo assim, a técnica de programação.
- **Função Main:** É a principal função do programa, responsável por chamar as sub-funções para realizar as suas determinadas tarefas. Dentro desta função foi declarada algumas variáveis, que posteriormente foram usadas para serem passadas por parâmetro em funções específicas. Encontraremos a chamada das seguintes funções: **menu_principal**, se o utilizador escolher a opção um (1) será executado o caso 1 onde encontraremos as seguintes funções, **cad_vacina** e **preencher_ri**, caso escolha a opção dois (2), será executado o caso 2, onde será chamada a função que irá mostrar as opções para ver os registos. Caso escolha a opção zero (0), o programa será encerrado.

2. CODIGO-FONTE

```
proj_reg_vac.c  ✕
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  #include <time.h>
5  #define max 5
6
7  typedef struct data{
8      int dia,mes,ano;
9  }DATA;
10
11  typedef struct comorb{
12      char comorbil[100];
13  } COMORB;
14
15  typedef struct Dados_Individual{
16      int cod_reg;
17      char nome_cmp[100];
18      char Doc_Ind[40];
19      char Num_Di[15];
20      char e_mail[30];
21      char Genero;
22      DATA dat_nasc;
23      int Num_TL;
24      COMORB comorb[3];
25      char nacional[30];
26      char prov_res[30];
27      char muni_res[30];
28  }DADOS;
29
30  typedef struct DADOS_RI{
31      char Nome_cp[100];
32      char DOC_IND[40];
33      char NUM_DI[15];
34      int COD_RI;
35  }DADOS_IDT;
36
37  void Interface(){
38      for (int i = 0; i < 50; i++)
39      {
40          printf("-");
41      }
42  }
43  void pausar_ex(){
44      printf("\n\nCLIQUE EM QUALQUER TECLA PARA CONTINUAR\n");
45      system("pause");
46      system("cls");
47  }
48  void inf_etapas(int etapa){
49      system("cls");
50      printf("\n\t\tREGISTO DE VACINA\n");
51      Interface();
52      printf("\nPREENCHE OS CAMPOS DA ETAPA %d DE 4\n", etapa);
53      Interface();
54  }
55  }
```

```

56 void sms_1_analise(int etapa){
57     printf("\nANALISANDO OS DADOS");
58     printf("\nAGUARDE");
59     for (int i = 0; i < 3; i++)
60     {
61         printf(".",i);
62         _sleep(1000);
63     }
64 }
65 void sms_2_sucesso(int etapa){
66     system("cls");
67     printf("\n\t\tREGISTO DE VACINA\n");
68     Interface();
69     printf("\nDADOS DA %d ETAPA SALVO COM SUCESSO",etapa);
70     _sleep(2000);
71     system("cls");
72 }
73 void sms_3_alert(){
74     char nao_exist[35]="OS DADOS SELECIONADO NAO EXISTE ";
75     printf("\n %s",strupr(nao_exist));
76     printf("\nOU A OPCAO ESCOLHIDA E INVALIDA\n\n");
77     Interface();
78     pausar_ex();
79 }
80 void sms_4_erro(int num){
81     char nao_exist[35]="deves digitar numeros de 1 a ";
82     printf("\n%s %d",strupr(nao_exist),num);
83     printf("\nDIGITE NOVAMENTE\n\n");
84     Interface();
85     pausar_ex();
86 }
87 void mod_1_reg(DADOS pessoa[],int i){
88     inf_etapas(1);
89     printf("\nDigite o Nome:");
90     fflush(stdin);
91     fgets(pessoa[i].nome_cmp, 100, stdin);
92
93     printf("Digite o Documento Individual:");
94     fflush(stdin);
95     fgets(pessoa[i].Doc_Ind, 40, stdin);
96
97     printf("Digite o Numero do documento Individual:");
98     fflush(stdin);
99     fgets(pessoa[i].Num_Di, 15, stdin);
100
101     printf("Digite o E_mail:");
102     fflush(stdin);
103     fgets(pessoa[i].e_mail, 30, stdin);
104
105     pessoa[i].cod_reg=(rand()%100)*12;
106     sms_2_sucesso(1);
107 }

```



```

108 void mod_2_reg(DADOS pessoa[],int i){
109     inf_etapas(2);
110     printf("\nDigite o numero de telefone: ");
111     scanf("%d",&pessoa[i].Num_TL);
112
113     for (; ){
114         system("cls");
115         inf_etapas(2);
116         printf("\nDigite a data de nascimento\n");
117         printf("Dia: ");
118         scanf("%d",&pessoa[i].dat_nasc.dia);
119         printf("mes: ");
120         scanf("%d",&pessoa[i].dat_nasc.mes);
121         printf("ano: ");
122         scanf("%d",&pessoa[i].dat_nasc.ano);
123         sms_1_analise(2);
124
125         if ((pessoa[i].dat_nasc.dia > 0 && pessoa[i].dat_nasc.dia <= 31) &&
126             (pessoa[i].dat_nasc.mes > 0 && pessoa[i].dat_nasc.mes <= 12) &&
127             ((pessoa[i].dat_nasc.ano>0) &&(2022-pessoa[i].dat_nasc.ano>=12) ))
128         {
129             sms_2_sucesso(2);
130             break;
131         }else{
132             printf("\nData incerta, nao pode fazer o registro");
133             _sleep(1500);
134             system("cls");
135             Interface();
136         }
137     }
138 }
139 void mod_3_reg(DADOS pessoa[],int i){
140     inf_etapas(3);
141     printf("\nDigite a Nacionalidade:");
142     fflush(stdin);
143     fgets(pessoa[i].nacional, 30, stdin);
144
145     printf("Digite Provincia:");
146     fflush(stdin);
147     fgets(pessoa[i].prov_res, 30, stdin);
148
149     printf("Digite a Municipio:");
150     fflush(stdin);
151     fgets(pessoa[i].muni_res, 30, stdin);
152     sms_2_sucesso(3);
153 }

```

```

154 void mod_4_reg(DADOS p[],int resp,int qtds_doe, int indice){
155     inf_etapas(4);
156     printf("\nPOSSUI COMORBILIDADE?");
157     printf("\n1- Sim");
158     printf("\n0- Nao\n");
159     scanf("%d", &resp);
160
161     switch (resp)
162     {
163     case 1:
164         for (;;)
165         {
166             system("cls");
167             inf_etapas(4);
168             printf("\nDigite o Numero de comorbilidade: ");
169             scanf("%d", &qtds_doe);
170             sms_1_analise(4);
171             inf_etapas(4);
172             if (qtds_doe>0 && qtds_doe<=3)
173             {
174                 for (int i = 1; i <= qtds_doe; i++)
175                 {
176                     printf("\nDigite a Comorbilidade numero %d: ",i);
177                     scanf("\n%[^\\n]s", &p[indice].comorb[i].comorbil);
178                 }
179                 sms_2_sucesso(4);
180                 break;
181             }else{
182                 sms_4_erro(3);
183             }
184         }
185         break;
186     case 0:strcpy(p[indice].comorb[1].comorbil,"sem comorbilidade");
187         break;
188     default:
189         break;
190     }
191 }
192 void CAD_VACINA(DADOS pessoa[],int ver_resp,int qtds_d,int i){
193     mod_1_reg(pessoa,i);
194     mod_2_reg(pessoa,i);
195     mod_3_reg(pessoa,i);
196     mod_4_reg(pessoa,qtds_d,ver_resp,i);
197
198     printf("\n\t\tREGISTO DE VACINA\n");
199     Interface();
200     printf("\nRegistro numero %d feito com sucesso", i);
201     _sleep(1300);
202 }

```



```

253 void op_mostrar( DADOS_IDT ver[], int op_most,int qtd_reg);
254 void main(){
255     int opc,c,op_most;
256     int qtd_reg,resp,qtds_doe;
257     DADOS_Cidadao[max];
258     DADOS_IDT aux[max];
259
260     srand(time(NULL));
261     for(;;){
262         system("cls");
263         Menu_principal();
264         scanf("%d",&opc);
265
266         switch(opc){
267             case 1:
268                 for ( ; ){
269
270                     printf("VAI REALISAR QUANTOS REGISTOS: ");
271                     scanf("%d", &qtd_reg);
272                     if(qtd_reg>0 && qtd_reg<=5){
273                         for (int i = 1; i <=qtd_reg; i++)
274                         {
275                             CAD_VACINA(Cidadao,resp,qtds_doe,i);
276                             preencher_RI(Cidadao,aux,i);
277                         }
278                         c=1;
279                         break;
280                     }
281                 else{
282                     system("cls");
283                     sms_4_erro(5);
284                 }
285             }
286             break;
287             case 2:
288                 if(c==1){
289                     system("cls");
290                     op_mostrar(aux,op_most,qtd_reg);
291                 }
292                 else{
293                     system("cls");
294                     printf("\n\tPRIMEIRO ESCOLHA A OPCAO 1\n");
295                     Interface();
296                     printf("\n");
297                     _sleep(1000);
298                     system("cls");
299                 }
300                 break;
301             case 0:
302                 system("cls");
303                 exit(0);
304                 break;
305             default:
306                 system("cls");
307                 printf("\nOpcao INVALIDA");
308                 _sleep(1000);
309             }
310         }
311 }

```


3. CONCLUSÃO

No princípio da elaboração deste trabalho foi levantada a seguinte pergunta: O que devemos fazer para se ter o control da população que irá tomar a vacina contra o virus da COVID-19? E diante da mesma chegou-se em diferentes respostas, mas no final desta pesquisa conseguimos cumprir com os nossos obetivos.

Depois de todo um processo árduo de investigação científico e metodológico, foi-nos possível chegar aos seguintes aspectos relacionados a execução do programa feito em linguagem c , usando estruturas de dados, funções variasves locas e formais, fazendo pasagens de parametro entre as funções, fois graças a estas e outras funções, que nos possibilitou a criação deste programa, agora já podemos realizar o registo da vacina contra a COVID-19 e obtermos o registo individual.

4. BIBLIOGRAFIA

Aguiar, Marcelo O. Introdução ao C em 10 aulas. / Marcelo Otone Aguiar;
Rodrigo Freitas Silva. - 1. ed. - Alegre: Marcelo Otone Aguiar, 2016