UNIVERSIDADE JEAN PIAGET DE ANGOLA

FACULDADE DE CIÊNCIAS E

TECNOLOGIAS

RELATÓRIO DE PROGRAMAÇÃO I

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

GRUPO A1

INTEGRANTES
Afonso Mambote
Beto Martins
Braulio Pereira
Lucrécio Daniel Barnabé
Luzolo Marcia da Costa Diango

Docente

Lúcia Tavira P. Bravo

Luanda, 2022

ÍNDICE

INTRODUÇÃO

Em virtude da pandemia provocada pelo virus da COVID-19, é obrigatório que as pessoas estejam vacinadas. Para o efeito, devem fazer o registo para a vacinação. Apois 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

Como podemos fazer um programa que realize de forma prática e ágil o registo da vacinação contra a COVID-19?

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 bibliotécas, 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 bibliotéca é a **string.h** serve para fazer manipulação de strings (conjutos de caracteres), e a ultima bibliotéca é a **time.h** que serve para manipular o tempo de procesamento 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 estruras 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 dara 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 par anaslisar se os dados foram atribuidos 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 possíbilita 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 selecionados não exite.
- **Função sms_4_erro:** foi criada para enviar uma mensagem de erro, quando o utilizador digitar um numero superior a 5 que é o limite máximo de pessoas que o programa pode registar, ou se o utilizador digitar um numero superios 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 numero que ele deve digitar.

No entanto, antes de realizar o registo geral, o grupo A1 pensou em divider 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 funões **inf_etapas**, e **sms_2_sucesso**. Está 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 e 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 posteriomente 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 uitlizador 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 regito 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:

- **Funcção preencher_ri**: Para prencher o ri foi utilizada a função strcpy que tem a função de copiar uma valor contido em uma variavel para outra varial. E tem a seguinte sintaxe: strcpy(distino,origem);
- **Mostrar_ri:** Tem a finalidade de mostrar todos o registos individuais cadastrados ao utilizador, tais como: **nome**, **codigo 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,opcões para mostrar o registo individual e sair ou terminar a execução.
- **Op_mostrar:** Mostra um meno 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 possivel, esta função foi declarada depois da função **main**, antes porém, foi criada um protótipo desta mesma função comprindo assim, a tecnica de programação.
- **Função Main:** É a princípal função do programa, responsavel por chamar as sub-funções para realizar as suas determindads tarefa. Dentro desta função foi declarada algumas variaveis, que posteriormente foram usadas para serem passadas por parametro em funções espcíficas. Encotraremos 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 escolhe a opção dois (2), será executado o caso 2,onde será chamada a funções que ira mostrar as opçõs para ver os registos. Caso escolha a opção zero (0), o programa será encerrado.

2. CODIGO-FONTE

proj_reg_vac.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#define max 5

typedef struct data{
    int dia,mes,ano;
}DATA;

typedef struct comorb{
    char  comorbil[100];
} COMORB;

typedef struct Dados_Individual{
    int     cod_reg;
    char nome_cmp[100];
    char  Doc_Ind[40];
    char  Num_Di[15];
    char  e_mail[30];
    char  Genero;
    DATA dat_nasc;
    int Num_TL;
    COMORB comorb[3];
    char  nacional[30];
    char  prov_res[30];
    char  muni_res[30];
}DADOS;

typedef struct DADOS_RI{
    char Nome_cp[100];
    char DOC_IND[40];
    char NUM_DI[15];
    int  COD_RI;
}DADOS_IDT;

void Interface(){
    for (int i = 0; i < 50; i++)
    {
        printf("-");
    }
}
void pausar_ex(){
    printf("\n\nCLIQUE EM QUALQUER TECLA PARA CONTINUAR\n");
        system("pause");
        system("cls");
}
void inf_etapas(int etapa){
    system("cls");
    printf("\n\t\tREGISTO DE VACINA\n");
    Interface();
    printf("\nPREENCHE OS CAMPOS DA ETAPA %d DE 4\n", etapa);
    Interface();

}
```

```c
void sms_1_analise(int etapa){
    printf("\nANALISANDO OS DADOS");
    printf("\nAGUARDE");
    for (int i = 0; i < 3; i++)
    {
        printf(".",i);
        _sleep(1000);
    }
}
void sms_2_sucesso(int etapa){
    system("cls");
    printf("\n\t\tREGISTO DE VACINA\n");
    Interface();
    printf("\nDADOS DA %d ETAPA SALVO COM SUCESSO",etapa);
    _sleep(2000);
    system("cls");
}
void sms_3_alert(){
        char nao_exist[35]="OS DADOS SELECIONADO NAO EXISTE ";
        printf("\n %s", strupr(nao_exist));
        printf("\nOU A OPCAO ESCOLHIDA E INVALIDA\n\n");
        Interface();
        pausar_ex();
}
void sms_4_erro(int num){
        char nao_exist[35]="deves digitar numeros de 1 a ";
        printf("\n%s %d", strupr(nao_exist),num);
        printf("\nDIGITE NOVAMENTE\n\n");
        Interface();
        pausar_ex();
}
void mod_1_reg(DADOS pessoa[],int i){
        inf_etapas(1);
        printf("\nDigite o Nome:");
           fflush(stdin);
           fgets(pessoa[i].nome_cmp, 100, stdin);

        printf("Digite o Documento Individual:");
           fflush(stdin);
           fgets(pessoa[i].Doc_Ind, 40, stdin);

        printf("Digite o Numero do documento Individual:");
           fflush(stdin);
           fgets(pessoa[i].Num_Di, 15, stdin);

        printf("Digite o E_mail:");
           fflush(stdin);
           fgets(pessoa[i].e_mail, 30, stdin);

        pessoa[i].cod_reg=(rand()%100)*12;
        sms_2_sucesso(1);
}
```

```
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 registo");
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 }
```

```c
void mod_4_reg(DADOS p[],int resp,int qtds_doe, int indice){
     inf_etapas(4);
    printf("\nPOSSUI COMORBILIDADE?");
          printf("\n1- Sim");
          printf("\n0- Nao\n");
          scanf("%d", &resp);

          switch (resp)
          {
          case 1:
              for (;;)
              {
                  system("cls");
                  inf_etapas(4);
                 printf("\nDigite o Numero de comorbilidade: ");
                 scanf("%d", &qtds_doe);
                 sms_1_analise(4);
                 inf_etapas(4);
                if (qtds_doe>0 && qtds_doe<=3)
                  {
                      for (int i = 1; i <= qtds_doe; i++)
                      {
                          printf("\nDigite a Comorbilidade numero %d: ",i);
                          scanf("\n%[^\n]s", &p[indice].comorb[i].comorbil);
                       }
                       sms_2_sucesso(4);
                       break;
                   }else{
                     sms_4_erro(3);
                   }
               }
              break;
          case 0:strcpy(p[indice].comorb[1].comorbil,"sem comorbilidade");
              break;
          default:
              break;
          }
}
void CAD_VACINA(DADOS pessoa[],int ver_resp,int qtds_d,int i){
        mod_1_reg(pessoa,i);
        mod_2_reg(pessoa,i);
        mod_3_reg(pessoa,i);
        mod_4_reg(pessoa,qtds_d,ver_resp,i);

        printf("\n\t\tREGISTO DE VACINA\n");
        Interface();
        printf("\nRegisto numero %d feito com sucesso", i);
        _sleep(1300);
}
```

```c
void preencher_RI(DADOS pessoa[], DADOS_IDT luc[],int i){
        strcpy(luc[i].Nome_cp,pessoa[i].nome_cmp);
        luc[i].COD_RI=pessoa[i].cod_reg;
        strcpy(luc[i].DOC_IND,pessoa[i].Doc_Ind);
        strcpy(luc[i].NUM_DI,pessoa[i].Num_Di);
}
void Mostrar_RI(DADOS_IDT pessoa[],int qtd_reg){

    char inf[40]="registos individuais a serem mostrados";

    printf("\n %d %s\n",qtd_reg,strupr(inf));
    Interface();
    for (int i = 1; i <= qtd_reg; i++)
    {
        printf("\n Nome: %s",strupr(pessoa[i].Nome_cp));
        printf("Codigo de registo: %dLLDDB%d", pessoa[i].COD_RI, pessoa[i].COD_RI*11 );
        printf("\nDocumento de ID: %s ", strupr(pessoa[i].DOC_IND));
        printf("Numero do DI: %s\n ", pessoa[i].NUM_DI);
         Interface();
    }pausar_ex();
}
void Menu_principal(){
    printf("\n\t\tESCOLHA A OPCAO\n");
    Interface();
    printf("\n1- Efetuar registo\n");
    printf("2- Opcoes Mostrar RI\n");
    printf("0- Sair\n");
}
void mostrar_Ri_por_parte(DADOS_IDT pessoa[], int opcao,int qtd_reg){
    system("cls");
        char inf[35]="dados do registo individual numero";
    Interface();
  for (int i = 1; i <= qtd_reg; i++){
       if (opcao==i)
        {
           printf("\n %s  %d\n",strupr(inf),opcao);
           Interface();
           printf("\nNome: %s",strupr(pessoa[i].Nome_cp));
           printf("Codigo de registo: %dLLDDB%d", pessoa[i].COD_RI,pessoa[i].COD_RI*11);
           printf("\nDocumento de I: %s", strupr(pessoa[i].DOC_IND));
           printf("Numero do Di: %s\n", pessoa[i].NUM_DI);
           Interface();
           pausar_ex();
        }
    }
      if((opcao>qtd_reg)){
           system("cls");
           sms_3_alert();
        }
}
```

```
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 }
```

```c
void op_mostrar( DADOS_IDT ver[], int op_most, int qtd_reg){
  for(;;){
    printf("\n\t\tESCOLHA A OPCAO \n");
    Interface();
    for (int i = 1; i <= qtd_reg; i++)
    {
        printf("\n%d - para mostrar o registo %d", i,i);

    }printf("\n6 - VER TODOS OS REGISTOS");
     printf("\n7 - MENU PRINCIPAL");
     printf("\n0 - SAIR\n");
     scanf("%d", &op_most);

     switch(op_most){
         case 1:
             mostrar_Ri_por_parte(ver,op_most,qtd_reg);
         break;
         case 2:
             mostrar_Ri_por_parte(ver,op_most,qtd_reg);
         break;
         case 3:
              mostrar_Ri_por_parte(ver,op_most,qtd_reg);
         break;
         case 4:
               mostrar_Ri_por_parte(ver,op_most,qtd_reg);
         break;
         case 5:
               mostrar_Ri_por_parte(ver,op_most,qtd_reg);
         break;
          case 6:
                system("cls");
                Mostrar_RI(ver,qtd_reg);
          break;
          case 7:
                system("cls");
                main();
          break;
          case 0:
                exit(0);
          break;
          default:
          system("cls");
           sms_3_alert();
        }
    }
}
```

3. CONCLUSÃO

No princípio da elaboração deste trabalho foi levantada a seguinte pergunta: Como podemos fazer um programa que realize de forma prática e ágil o registo da vacinação contra a COVID-19? E diante da mesma chegou-se em diferentes respostas, mas no final desta pesquisa conseguimos cumprir com os nossos objetivos.

Depois de todo processo árduo de investigação cientifica e metodológico, foi-nos possível chegar aos seguintes aspectos relacionados a execução do programa feito em linguagem c , usando estruturas de dados, funções variáves locais e formais, fazendo pasagens de parametro entre as funções, foi 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