```c
/* GRUPO: Panelinha
Membros:
Henrique Soares Costa (Matricula : 20213010852)
João Vitor Araujo Leão (Matricula : 20213008059)
Lucas Silva Moreira (Matricula : 20213008101)
Roginaldo Reboucas Rocha Junior (Matricula : 20213008157)
Pedro Carneiro Rabetim (Matricula : 20213008139)
Gabriel Henrique Martins (Matricula : 20213009619)

CASO QUEIRA VER MAIS INFORMACOES:
github.com/riquetret/Trabalho-De-Final-de-Prog
*/

#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <time.h>
#include <locale.h>
#include <ctype.h>
#include <time.h>

typedef struct  //Declara nosso struct de filmes
{
    int identificador;
    char nome[50];
    char genero[30];
    unsigned short int anoLancamento;
    char nomeDiretor[30];
}Filme;

enum Estados{   //Declara nossos estados do programa (Como se fosse uma FSM, ou maquina de estados)
    Saida=0,
    Adicionar=1,
    Editar=2,
    Remover=3,
    Exibir=4,
    Gravar=5,
    Ler_Banco=6,
};
//Declara nossos protótipos
void limpa_tela();                                                  //Limpa a tela da console
void limpa_buffer();                                                //Limpa o buffer do teclado (lixo do teclado)
int le_numero(void *numero,double ri, double rf, char t);           //Le um numero no intervalo ri até rf, ou seja, ri<numero<rf
void inicializa_arquivo(FILE **ptr,char *nome,char *modo);          //Inicializa nossos ponteiros de arquivo no modo desejado

int adicionaFilme(Filme *ptr, int tam,FILE *ptr2);
int geraIdentificador(FILE *ptr);                                   //Gera identificador
int editaFilme(Filme *ptr,int posicao,FILE *ptr2);
int removeFilme(Filme *ptr,int posicao);
int imprimeFilmes(Filme *ptr,int posicao);
int escreveFilmes(Filme *ptr,FILE *arq_salvar,int posicao,int *tam);//Escreve do vetor de filmes para o banco de dados
int gravaFilmes(Filme *ptr,FILE *arq_orig,int *tam);                //Grava os filmes do vetor para o banco de dados
int leFilmes(Filme *ptr,int acao,FILE *dados);                      //Ler os filmes do banco de dados e atribuir ao vetor
int buscaFilme(Filme *ptr,FILE *ptr2,char *nome_filme);             //Busca o nome do filme desejado


int main(){
    Filme filmes_lidos[50];
    //Cria Vetor de Structs
    removeFilme(&filmes_lidos[0],-1);
    //Vamos resetar o vetor de filmes do programa
    enum Estados opcao;
    //Declara Enum com estados
```

```
61
62      char menu[]=
        //Declara exibição padrão do menu
63      "0)Sair do Programa\
64      \n1)Adicionar Filmes\
65      \n2)Editar Filmes\
66      \n3)Remover Filmes\
67      \n4)Exibir Filmes\
68      \n5)Gravar Filmes\
69      \n6)Ler Filme do Banco";
70
71      char acoes[][9]={"editar?","deletar?"};
        //Declara matriz para não repetir o código basicamente
72
73      char filme_desejado[50];
        //Vetor para receber o nome do filme desejado/a ser buscado
74
75      int posicoes,filmes_adicionados=0,retorno;
        //Declara posicoes (Para alterar uma posição do vetor), declara
        filmes_adicionados (para contabilizar quantos filmes já foram adicionados) e por
        fim declara retorno para analisar o retorno de algumas funções
76
77      char status_do_banco[]="NAO";
        //O banco de dados esta aberto? Neste caso nao
78
79      FILE *arquivo;
        //Declara ponteiro de arquivo
80      //setlocale(LC_ALL, "Portuguese");
81
82      inicializa_arquivo(&arquivo,"filmes.txt","r+");
        //Abre o arquivo "filmes.txt" no modo leitura para atualizar
83
84      do
85      {
86          printf("================================================================\n");
87          printf("BEM-VINDO ao forum MANIA-FILMES\n");
                                                //Mensagens iniciais
88          printf("\nSe sua entrada nao for processada, aperte \"enter\" DUAS vezes\n\n"
            ); //Devido ao limpa_buffer que pode tentar ler quando nao ha entradas no
            buffer do teclado
89
90          printf("Voce carregou filmes do banco de dados? %s",status_do_banco);
                        //Exibe se temos um filme carregado do banco de dados
91          printf("\nQuantos filmes voce ja carregou no sistema? %d Filmes (Max 50
            Filmes)\n",filmes_adicionados); //Mostra quantos filmes foram adicionados
92
93          printf("\n%s\nDigite qual opcao deseja: ",menu);
            //Exibe opções ao usuário
94          le_numero(&opcao,0,6,'i');
            //Le opção escolhida do menu (char menu[])
95
96          limpa_tela();
            //Limpa a tela
97          if(1<opcao && opcao<4){
            //A pessoa deseja editar? ou deletar ou exibir?
98              imprimeFilmes(&filmes_lidos[0],-1);          //Exiba os filmes cadastrados
99              printf("\nQual posicao deseja %s (0 para cancelar)\n",acoes[opcao-2] );
                   //Pergunta qual posicao deseja editar ou deletar ou (Sabemos que a
                pessoa quer uma dessas três opcoes)
100             le_numero(&posicoes,0,50,'i');                                  //Le a
                posição desejada
101             posicoes--;
                //Decrementa posicoes para escolher a posicao correta no vetor
102             if(posicoes<0)continue;                                         //Se
                deseja cancelar saia e volte para o loop
103         }
104
105         switch(opcao){
            //Analisando a escolha da pessoa
106             case Adicionar:
107                 retorno = adicionaFilme(&filmes_lidos[0],filmes_adicionados,arquivo);
                           //retorno recebe a quantidade filmes adicionados
108                 if(retorno==-1)printf("ERRO: O sistema nao suporta a adicao de mais
```

```
                filmes");//Vetor cheio?
109             else{
110                 filmes_adicionados+=retorno;
                    //Incrementa os filmes_adicionados
111                 printf("\nFilmes adicionados com sucesso\n\n");
                    //Mostra mensagem ao usuário
112             }
113         break;
114         case Editar:
115             limpa_tela();
                //Limpa a tela
116             imprimeFilmes(&filmes_lidos[0],posicoes);
                //Exibe filme escolhido
117             editaFilme(&filmes_lidos[0],posicoes,arquivo);
                                            //Edita filme escolhido
118         break;
119         case Remover:
120             removeFilme(&filmes_lidos[0],posicoes);
                //Remove Filme Escolhido
121             if(filmes_adicionados>0)filmes_adicionados--;
                //Tiramos um filme logo decremente filmes_adicionados
122             printf("Filme removido com sucesso\n");
                //Mostra Filme Removido com sucesso
123         break;
124         case Exibir:
125             printf("Qual posicao deseja exibir? -1 para todos\n");
                //Pergunta qual posicao deseja imprimir/exibir
126             le_numero(&posicoes,-1,50,'i');
                //Exibe a posicao
127             posicoes--;
                //Decrementa posicoes, para escolher a posicao correta no vetor
128             imprimeFilmes(&filmes_lidos[0],posicoes);
                //Exibe filme escolhido ou todos se posicoes igual a -1
129         break;
130         case Gravar:
131             printf("ALERTA: A gravacao de filmes, retira da memoria os filmes ja
                carregados\n\n");
132             gravaFilmes(&filmes_lidos[0],arquivo,&filmes_adicionados);
                //Grava os filmes adicionados ou alterados
133             strcpy(status_do_banco,"NAO");
                //Tudo foi resetado, logo o banco nao esta mais carregado
134         break;
135         case Ler_Banco:
136             limpa_tela();
                //limpa a tela
137             printf("ALERTA: A leitura do banco de dados, sobrescreve os filmes
                ja carregados na memoria\n\n");
138             printf("Voce deseja:\n");
139             printf("0)Cancelar a operacao\
140                     \n1)Procurar Por Um Filme\
141                     \n2)Abrir os filmes do Banco de dados\
142                     \n3)Deletar/Recriar o Banco de Dados\n");
143             printf("Sua opcao: ");
144             le_numero(&posicoes,0,3,'i');                                   //Le
                a opcao desejada
145             switch (posicoes)
                //Analisa a escolha
146             {
147             case 1:
                                                                                //A
                pessoa deseja procurar um filme
148                 while(1){
149                     printf("\nDigite o filme desejado: ");
                                                //Pede o filme
150                     fgets(filme_desejado,50,stdin);
                                                    //Le o filme para procurar
151                     limpa_buffer();
                                                                    //Limpa
                        lixo do teclado
152                     filme_desejado[strcspn(filme_desejado, "\n")] = 0;
                                //Remove \n lido pelo fgets
153                     for(posicoes=0;filme_desejado[posicoes]!='\0';posicoes++)
                        filme_desejado[posicoes]=tolower(filme_desejado[posicoes]);
```

```
                        //Transforma o nome do filme digitado para minusculo
154                     if(buscaFilme(&filmes_lidos[0],arquivo,filme_desejado)==-1){
                            //Se o filme nao foi encontrado
155                         printf("\nErro: Filme não encontrado\n");
                                                //Exiba o erro
156                         printf("Deseja continuar a busca? (Digite S para sim e N
                            para nao)\n"); //Pergunte se deseja continuar a busca
157                         filme_desejado[0]=getchar();
                                                            //Leia a opcao digitada
158                         filme_desejado[0]=tolower(filme_desejado[0]);
                                                //Transforme a opcao digitada para
                            minusculo
159                         if(filme_desejado[0]=='n')break;
                                                        //Se pessoa deseja sair,
                            Saia do loop
160                     }
161                     else{
162                         filmes_adicionados++;
                                                                    //Se o filme foi
                            encontrado, logo incremente filmes_adicionados
163                         strcpy(status_do_banco,"SIM");
                                                        //Fale que ha filmes
                            carregados do banco
164                         break;

                            //Se o filme foi encontrado, saia do loop
165                     }
166                 }//END while(1)
167                 break;
168             case 2:
                //A pessoa deseja ler o banco de dados
169                 retorno=leFilmes(&filmes_lidos[0],2,arquivo);
                    //O leFilmes busca os primeiros 50 filmes do banco e retorna a
                    quantidade adicionada para retorno
170                 filmes_adicionados+=retorno;
                    //Incremene o filmes_adicionados
171                 if(retorno>0)strcpy(status_do_banco,"SIM");
                    //Se algum filme foi salvo, fale que que ha filmes carregados do
                    banco
172                 break;
173             case 3:
174                 fclose(arquivo);
                    //Fecha o arquivo do banco de dados por seguranca
175                 remove("filmes.txt");
                    //Remove o banco de dados
176                 inicializa_arquivo(&arquivo,"filmes.txt","r+");
                    //Recria o banco de dados "filmes.txt" no modo leitura para
                    atualizar
177                 break;
178             default:
179                 break;
180             }
181             break;//END CASE LER BANCO
182         } //END SWITCH(OPCAO)
183     }while(opcao);//END WHILE(1)
184
185     fclose(arquivo);
        //Salva o arquivo para encerrar as operacoes
186     return 0;
187 }//END MAIN
188
189 /*
190 Função: limpa_tela
191 Autor: Feita por Henrique Soares Costa, github.com/RIQUETRET
192 Entradas: Nenhuma
193 Saídas: Sua tela limpa como cristal
194 Retorno: Nenhum
195
196 Objetivo: Apaga a tela da sua console
197 usando comandos tipicos do windows e do linux
198 após isso da 2 quebras de linha
199 */
200 void limpa_tela(){
```

```c
    system("cls");
    //Vamos apagar as mensagens iniciais
    system("clear");
    //Vamos apagar as mensagens iniciais
    printf("\n\n");
    //Vamos apagar as mensagens iniciais
}
/*
Função: Limpa_Buffer
Autor: Baseado em outras pessoas, modificada por Henrique.
Entradas:Stdin
Saídas: Nenhuma
Retorno: Nenhum

Objetivo: Consumir caracteres
adicionais presentes no stdin.
Se a função encontrar um EOF ela
Reseta o stdin para futuras leituras
*/
void limpa_buffer() {
    char caracter=0;                                    //Declara char para a leitura
    do {
    caracter = fgetc(stdin);                            //Le caracter por caracter
    ate "zerar" stdin
    } while (caracter != '\n' && caracter!=EOF);        //Se foi encontrado uma
    quebra de linha ou um erro saia
    if(caracter==EOF)clearerr(stdin);                   //Se foi encontrado um EOF,
    resete stdin
}

/*
Função: le_numero
Autor: Feita por Henrique Soares Costa, github.com/RIQUETRET
Entradas: Ponteiro para o número,intervalos ri e rf, tipo t
Saídas: o *numero recebe um valor lido pelo teclado
Retorno: Retorna 0 se tudo ocorreu bem

Objetivo: Lê o teclado e atribui ao *numero,
contudo o valor lido deve estar entre ri e rf (ri<=valor lido<=rf).
É importante especificar o tipo de variavel com o char t
(se t='d', leia um double do teclado,t='f'' lê float, t='i' lê int,
 caso nenhum desses seja satisfeito ele irá ler um char do teclado)
*/
int le_numero(void *numero,double ri, double rf, char t){
    int erro=-1;

    //Temos inicialmente um erro pq nenhuma leitura foi realizada
    do
    {
        switch (t)

        //Analisa o tipo de variavel a ser lido
        {
        case 'd':

        //Variavel do tipo double
            erro=scanf("%lf",(double *) numero);
                                                        //Le um double e erro recebe
            a quantidade de informacoes lidas
            if(erro==1){

            //Conseguimos ler o numero?
                if( (*(double *) numero) >= ri &&
                                                            //Esse numero lido esta
                dentro o intervalo especificado?
                (*(double *) numero) <= rf ) continue;
                                                    //Se sim pule para o proximo
                loop e finalize
                else erro=0;
                                                                                //Se
                nao defina um erro
            }
            break;
```

```
252             case 'f':

                //Variavel do tipo float
253                 erro=scanf("%f",(float *) numero);
                                                                //Le um double e erro
                    recebe a quantidade de informacoes lidas
254                 if(erro==1){

                    //Conseguimos ler o numero?
255                     if( (*(float *) numero) >= (float) ri &&
                                                            //Esse numero lido esta dentro o
                        intervalo especificado?
256                         (*(float *) numero) <= (float) rf ) continue;
                                                        //Se sim pule para o proximo loop e
                        finalize
257                         else erro=0;
                                                                                    //Se
                        nao defina um erro
258                 }
259                 break;
260             case 'i':

                //Variavel do tipo inteira
261                 erro=scanf("%d",(int *) numero);
                                                                //Le um double e erro
                    recebe a quantidade de informacoes lidas
262                 if(erro==1){

                    //Conseguimos ler o numero?
263                     if( (*(int *) numero) >= (int) ri &&
                                                            //Esse numero lido esta
                        dentro o intervalo especificado?
264                         (*(int *) numero) <= (int) rf ) continue;
                                                        //Se sim pule para o proximo loop
                        e finalize
265                         else erro=0;
                                                                                    //Se
                        nao defina um erro
266                 }
267                 break;
268             default:

                //Variavel do tipo char
269                 erro=scanf("%c",(char *) numero);
                                                                //Le um double e erro
                    recebe a quantidade de informacoes lidas
270                 if(erro==1){

                    //Conseguimos ler o char?
271                     if( (*(char *) numero) >= (char) ri &&
                                                            //Esse numero lido esta dentro
                        o intervalo especificado?
272                         (*(char *) numero) <= (char) rf ) continue;
                                                        //Se sim pule para o proximo loop e
                        finalize
273                         else erro=0;
                                                                                    //Se
                        nao defina um erro
274                 }
275                 break;
276             }//END switch(t)
277             limpa_buffer();
                                                                                //Limpa
                lixo do teclado se errar
278             printf("\nErro Digite Novamente: ");
                                                            //Mostra mensagem de erro
                caso erre
279         } while (erro<=0);
                                                                                //Houve
            algum erro?
280         limpa_buffer();
                                                                                //Limpa
            lixo do teclado se errar
```

```
281         return 0;

            //Retorna 0, indicando sucesso
282     }
283
284     /*
285     Função: inicializa_arquivo
286     Autor: Feita por Henrique Soares Costa, github.com/RIQUETRET
287     Entradas: Seu ponteiro de arquivo (**arq), nome do seu arquivo(*nome), modo para
        abrir (*modo)
288     Saídas: Seu ponteiro de arquivo
289     Retorno: Nenhum
290
291     Objetivo: Abri o arquivo *nome no modo leitura,
292     caso o arquivo não exista, a função cria o arquivo.
293     Posteriomente a função reabre o arquivo no modo especificado (*modo)
294     */
295     void inicializa_arquivo(FILE **arq,char *nome,char *modo){
296         *arq=fopen(nome,"r");                                   //Tente abrir o
            arquivo inicialmente no modo leitura
297         if(*arq==NULL){                                         //Se ele não existir
            ou o programa não ter permissão suficiente, crie ele
298             fclose(*arq);                                       //Feche o arquivo
                por segurança
299             *arq=fopen(nome,"w");                               //Tente então criar
                o arquivo
300             if(*arq==NULL)exit(EXIT_FAILURE);                   //Se a criação nao
                foi possível finalize o programa e exiba erro
301         }
302         fclose(*arq);                                           //Fecha o arquivo
            por segurança
303         *arq=fopen(nome,modo);                                  //Reabre o arquivo
            no modo selecionado
304     }
305
306     /*
307     Função: adicionaFilme
308     Autor: Lucas Silva e Roginaldo
309     Entradas: Seu vetor de filmes (*ptr), quantos filmes ja foram adicionados ao vetor
        (tam)
310     e sua base de dados (*ptr2)
311     Saídas: Seu vetor cheio de filmes
312     Retorno: Quantidade de filmes adicionados
313
314     Objetivo: Pede quantos filmes deseja adicionar, verifica quantos filmes ja tenho no
        vetor,
315     e por fim salva no vetor o filme desejado utilizando como suporte o edita filmes
316     */
317
318     int adicionaFilme(Filme *ptr, int tam,FILE *ptr2){ //desenvolvimento do prototipo da
        funcao adiciona filme
319         if(tam>=50)return -1;   //Se o vetor ja estiver cheio retorne erro
320         int qt_filmes, i;       //Declara qt_filmes = quantidade filmes e nosso iterador i
321         printf("Quantos filmes deseja adicionar? (Digite zero para cancelar a
            operacao)\n"); //solicita quantidades de filmes a serem adicionados no vetor
322         le_numero(&qt_filmes,0,50,'i');                         //Le a quantidade de filmes
            que o usuario deseja (De 1 Filme até 50 Filmes)
323
324         if (qt_filmes==0)return 0;                              //Se a pessoa nao quiser
            adicionar filmes, volte ao programa principal
325         else if(qt_filmes+tam>50){                              //A pessoa deseja adicionar
            mais do que o vetor suporta?
326             qt_filmes=50-tam;                                   //Defina então o máximo para
                ser adicionado, ou seja, quantos espaços vazios eu tenho no vetor para
                adicionar
327             printf("\nALERTA O maximo suportado para adicao de filmes eh:%d\n\n",
                qt_filmes);  //Exibe alerta para a pessoa ter ciência
328         }
329
330         tam=0;                  //O Tam agora tem nova funcionalidade como indicador de
            quantos filmes ja foram adicionados de fato no vetor
331         for(i=0;i<50;i++){      //Vamos adicionar os filmes no vetor percorrendo as 50
            posicoes
```

```
332             if (ptr[i].identificador!=0) continue;  //Caso haja um filme já cadastrado
                na posicao de memoria pule para a proxima posicao
333             if((tam==qt_filmes) || (editaFilme(&(*ptr),i,ptr2)==-1) )break;
                                    //Adiciona o filme a partir do edita filme e caso a pessoa
                queira cancelar sai do loop, tambem faz validacao caso a pessoa tenha
                digitada a quantidade de filmes desejada
334             tam++;              //Se a pessoa introduziu um filme incremente
335         }
336        return tam; //Retorna a quantidade filmes adicionada
337     }
338     /*
339     Função: geraIdentificador
340     Autor: Feita por Henrique Soares Costa, github.com/RIQUETRET
341     Entradas: O ponteiro de arquivo da sua base de dados (*ptr)
342     Saídas: Nenhum
343     Retorno: Um identificador unico para seu filme
344
345     Objetivo: Gera um numero aleatorio e compara se ha algum identificador igual
346     a este numero gerado presente na base de dados
347     */
348     int geraIdentificador(FILE *ptr){
349         srand(time(NULL));                                      //Gera semente para
                funcao rand()
350         int numerado_gerado,identificador_lido;                 //Para receber o numero
                gerado e para receber o identificador da base de dados
351         char lido[55];                                          //Vetor para ler as
                linhas do arquivo
352         numerado_gerado = rand()+1;                             //Vamos gerar um numero
                de 1 até RAND_MAX+1
353         fseek(ptr,0,SEEK_SET);                                  //Reposiciona ponteiro
                de arquivo no inicio do meu arquivo "filmes.txt"
354         while (fgets(lido,0,ptr)!=NULL)
355         {
356             if (lido[0]='i')                                    //A linha lido no
                arquivo é a linha de um identificador?
357             {
358                 sscanf(lido,"%[i:]%d",identificador_lido);      //Vamos ler o
                    identificador em asc2 e atribuir a variavel identificador_lido
359                 if(identificador_lido==numerado_gerado)numerado_gerado = rand()+1;//Se
                    encontramos um identificador igual na base de dados, gere um novo
                    identificador para o filme
360             }
361         }
362         return numerado_gerado;                                 //Retorna o numero_gerado
363     }
364     /*
365     Função: editaFilme
366     Autor: João Vitor Araujo Leao
367     Entradas: O ponteiro de arquivo da sua base de dados (*ptr),
368     a posicao desejada do vetor para editar (posicao), e seu banco de dados (*ptr2)
369     Saídas: Seu vetor editado com os filmes
370     Retorno: 0 se sucesso e -1 se erro
371
372     Objetivo: Pergunta para o usuario as informacoes do filme
373     posteriormente gera um identificador para este filme
374     se ele nao tiver um.
375     */
376     int editaFilme(Filme *ptr,int posicao,FILE *ptr2){
377         int i;
378         printf("Caso queira cancelar a operacao como um todo, digite \"0\" em pelo menos
                um campo a seguir\n\n");
379
380         printf("Digite o nome do %d%c filme: ", posicao,248); //atrelando o nome do
                filme inserido pelo usuario ao filme colocado no vetor
381         fgets(ptr[posicao].nome,50,stdin);    //Le o nome do filme com fgets para nao
                deixar overflow
382         ptr[posicao].nome[strcspn(ptr[posicao].nome, "\n")] = 0;    //Remove \n lido
                pelo fgets
383         limpa_buffer();                 //Remove lixo do teclado
384         if(strcmp(ptr[posicao].nome,"0")==0)return -1;          //Se a pessoa digitou 0
                sai do programa e cancele
385
386
```

```
387         printf("Digite o genero do %d%c filme: ", posicao,248); //atrelando o genero do
            filme inserido pelo usuario ao filme colocado no vetor
388         fgets(ptr[posicao].genero,30,stdin);  //Le o genero com fgets para nao deixar
            overflow
389         ptr[posicao].genero[strcspn(ptr[posicao].genero, "\n")] = 0;    //Remove \n lido
            pelo fgets
390         limpa_buffer();
391         if(strcmp(ptr[posicao].genero,"0")==0)return -1;        //Se a pessoa digitou 0
            sai do programa e cancele
392
393         printf("Digite o ano de lancamento do %d%c filme", posicao,248); //atrelando o
            ano de lancamento do filme inserido pelo usuario
394         printf("\nDigite um ano entre 1900 e 2021: ");
395         le_numero(&(ptr[posicao].anoLancamento),1900,2021,'i');   //Le ano do filme com
            fgets para nao deixar overflow
396         limpa_buffer();
397
398         printf("Digite o nome do diretor do %d%c filme: ", posicao,248); //atrelando o
            nome do diretor do filme inserido pelo usuario
399         fgets(ptr[posicao].nomeDiretor,30,stdin); //Le o nome do diretor com fgets para
            nao deixar overflow
400         ptr[posicao].nomeDiretor[strcspn(ptr[posicao].nomeDiretor, "\n")] = 0;  //Remove
            \n lido pelo fgets
401         limpa_buffer();
402         if(strcmp(ptr[posicao].nomeDiretor,"0")==0)return -1;   //Se a pessoa digitou 0
            sai do programa e cancele
403
404         for(i=0;ptr[posicao].nome[i]!='\0';i++)ptr[posicao].nome[i]=tolower(ptr[posicao].
            nome[i]); //Transforma as letras para minusculo
405         for(i=0;ptr[posicao].genero[i]!='\0';i++)ptr[posicao].genero[i]=tolower(ptr[
            posicao].genero[i]); //Transforma as letras para minusculo
406         for(i=0;ptr[posicao].nomeDiretor[i]!='\0';i++)ptr[posicao].nomeDiretor[i]=tolower
            (ptr[posicao].nomeDiretor[i]); //Transforma as letras para minusculo
407
408         if(ptr[posicao].identificador==0){                      //Se o filme nao tiver
            identificador
409             ptr[posicao].identificador=geraIdentificador(ptr2);         //Adicione um
                identificador a ele, analisando obviamente a base de dados
410         }
411         return 0;
412     }
413
414     /*
415     Funcao: removeFilme
416     Autor: Lucas Silva e Roginaldo Junior
417     Entradas: O seu vetor de filmes (*ptr) e a posicao que deseja
418     remover (posicao), obs: se posicao<0, remove todo o vetor de filmes
419     Saídas: Seu vetor com alguns elementos a menos
420     Retorno: 0 se sucesso
421
422     Objetivo: remover o filme que esta na posicao inserida como
423     parametro da funcao, 'zerando' o identificador e o ano de lancamento
424     atribuindo o valor 0 e os demais dados atribuindo o caracter '\0' na
425     primeira posicao da string.
426     */
427
428     int removeFilme(Filme *ptr, int posicao){ //desenvolvimento do prototipo da funcao
        remove filme
429         int i = 0;
430
431         if (posicao<0) posicao = 50;                            //Se posicao<0, vamos apagar
            então os 50 elementos do vetor
432         else{                                                   //Caso contrario, vamos
            apagar apenas um elemento do vetor
433             i=posicao;                                          //Vamos posicionar o
                iterador na posicao escolhida
434             posicao++;                                          //E vamos fazer o loop for
                com apenas 1 loop, logo posicao++
435         }
436
437         for(;i<posicao;i++){                                    //Percorrendo o vetor
438                 ptr[i].identificador = 0;                       //removendo o identificador
                    atrelado ao filme zerando ele
```

```
439             strcpy(ptr[i].nome,"");                         //removendo o nome atrelado
                ao filme zerando ele
440             strcpy(ptr[i].genero,"");                       //removendo o genero
                atrelado ao filme zerando ele
441             ptr[i].anoLancamento = 0;                       //removendo o ano de
                lancamento atrelado ao filme zerando ele
442             strcpy(ptr[i].nomeDiretor,"");                  //removendo o nome do
                diretor atrelado ao filme zerando ele
443         }
444         return 0; //retorno da funcao apos sua conclusao
445     }
446
447     /*
448     Funcao: ImprimeFilme
449     Autor: Pedro Carneiro Rabetim
450     Entradas: O seu vetor de filmes (*ptr) e a posicao que deseja
451     imprimir (posicao), obs: se posicao<0, imprime todo o vetor de filmes
452     Saídas: Impressao na tela do seu vetor
453     Retorno: 0 se sucesso
454     Objetivo: Analisa a posicao introduzida na funcao
455     e imprime o filme correspondente
456     */
457
458     int imprimeFilmes(Filme *ptr,int posicao){
459
460         int n=0,parada=50;                                                  //Define
            inicialmente iterado no inicio do vetor(0) e parada no fim desse vetor (De 0 ate
            49 posicoes)
461         if(posicao >= 0){                                                   //A pessoa
            deseja imprimir uma posicao especifica?
462             n=posicao;                                                      //O iterador
                ficara nesta posicao
463             parada=posicao+1;                                               //E vamos
                executar mais um loop, logo posicao++
464         }
465         for(;n<parada;n++){                                                 //Vamos
            percorrer o vetor ate parada
466             if(ptr[n].identificador!=0){                                    //O filme
                tem um identificador valido?
467                 printf("\n==================");                           //Imprime
                    caracteristicas do filme
468                 printf("\nNome:");
469                 puts(ptr[n].nome);
470                 printf("Genero:");
471                 puts(ptr[n].genero);
472                 printf("Ano:");
473                 printf("%d\n",ptr[n].anoLancamento);
474                 printf("Diretor:");
475                 puts(ptr[n].nomeDiretor);
476                 printf("Posicao:%d",n+1);
477                 printf("\n==================\n");
478             }
479         }
480         return 0;                                                           //Retorna 0
481     }
482     /*
483     Função: escreveFilmes
484     Autor: Feita por Henrique Soares Costa, github.com/RIQUETRET
485     Entradas: Seu vetor de filmes (*ptr), a sua nova base de dados (*arq_salvar),
486     a posicao do vetor que deseja salva na base de dados (posicao), a quantidade
487     de filmes que ja existem no vetor (*tam)
488     Saídas: Uma base dados cheianha de filmes
489     Retorno: Zero se tudo ok
490
491     Objetivo: Coloca as informacoes do vetor na nova base de dados
492     escrevendo corretamente. É imporante salientar que apos salvo
493     o filme na base dados, ele é deletado do vetor e tam tem seu conteudo
494     decrementado
495     */
496     int escreveFilmes(Filme *ptr,FILE *arq_salvar,int posicao,int *tam){
497         fprintf(arq_salvar,"i: %d\n",ptr[posicao].identificador);          //Coloca o
            identificador do nosso filme na base de dados
498         fprintf(arq_salvar,"n: %s\n",ptr[posicao].nome);                   //Coloca o
```

```
        nome do nosso filme na base de dados
499     fprintf(arq_salvar,"g: %s\n",ptr[posicao].genero);                  //Coloca o
        gernero do nosso filme na base de dados
500     fprintf(arq_salvar,"l: %d\n",ptr[posicao].anoLancamento);           //Coloca o ano
        de lancamento do nosso filme na base de dados
501     fprintf(arq_salvar,"d: %s\n",ptr[posicao].nomeDiretor);             //Coloca o
        nome do diretor do nosso filme na base de dados
502
503     removeFilme(&(*ptr),posicao);                                       //Vamos agora
        apagar o filme salvo no vetor
504     *tam=(*tam)-1;                                                      //Ja que
        tiramos um filme, vamos então decrementar o indicador da quantidade de filmes
        salvas no vetor
505     return 0;                                                           //Retorna 0
506 }
507 /*
508 Função: gravaFilmes
509 Autor: Feita por Henrique Soares Costa, github.com/RIQUETRET
510 Entradas: Seu vetor de filmes (*ptr), seu banco de dados original(*arq_orig),
511 a quantidade de filmes salva no seu vetor(*tam)
512 Saídas: Um novo arquivo "filmes.txt" atualizado
513 Retorno: Zero se tudo ok
514
515 Objetivo: Cria uma copia de "filmes.txt" atualizando a copia com os valores
516 do vetor de filmes. Feito isso "filmes.txt" é deletado e a copia torna-se
517 "filmes.txt".
518 */
519 int gravaFilmes(Filme *ptr,FILE *arq_orig,int *tam){
520     if(*tam==0)return 0;                                                //O nosso vetor
        de filmes está vazio? Entao nao precisa gravar nada
521     FILE *arq_dst=fopen("filmes_copia.txt","w");                        //Cria uma copia
        para escrever os dados
522     char lido[55];                                                      //Vetor para ler
        as linhas do "filmes.txt"(*arq_orig)
523     int j;                                                              //Declara
        iterador j
524     char *erro;                                                         //Ponteiro para
        receber erro do fgets
525     int identidade;                                                     //Variavel para
        receber o identificador dos filmes
526     fseek(arq_orig,0,SEEK_SET);                                         //Posiciona
        ponteiro para o inicio de "filmes.txt"
527
528     do{
529         erro=fgets(lido,55,arq_orig);                                   //Vamos ler uma
            linha do meu arquivo
530         if(erro==NULL){                                                 //Fim do banco
            de dados encontrado?
531             for(j=0;j<50 && *tam!=0;j++){                               //Vamos entao
                salvar o nosso vetor no fim "filmes.txt" ate acabar...
532                 if(ptr[j].identificador!=0)escreveFilmes(&(*ptr),arq_dst,j,&(*tam));
                       //Se o identificador no vetor de filmes nao eh nulo, logo escreva
                    este filme no meu arquivo de destino "filmes_copia.txt"
533             }
534         }
535         else{                                                           //Se meu banco
            de dados nao acabou...
536             if(lido[0]=='i'){                                           //Se a linha
                lida do banco foi a linha de um identifcador, logo...
537                 sscanf(&(lido[3]),"%d",&identidade);                    //Vamos
                transformar o identificador em ASC2 para um inteiro usando sscanf
538                 for(j=0;j<50;j++){                                      //Agora vamos
                percorrer o vetor e verificar se este identificador ja existe
539                     if(identidade==ptr[j].identificador){               //Se o
                    identificador lido do arquivo for igual ao do vetor
540                         identidade=-1;                                  //Defina
                        identidade -1 (para analises adiante)
541                         break;                                          //Saia do loop
542                     }
543                 }//END for(j=0;j<50;j++)
544                 if(identidade==-1){                                     //Se foi
                encontrado um identificador igual ao presente no vetor, faça a
                escrita deste filme
```

```
545                     escreveFilmes(&(*ptr),arq_dst,j,&(*tam));       //Vamos então
                        escrever o filme do vetor encontrado, no arquivo
546                     for(j=0;j<4;j++)fgets(lido,55,arq_orig);        //Vamos ignorar
                        as proximas 4 linhas pois o filme do meu vetor ja foi escrito
547                     continue;                                       //Ja que
                        terminos essa analise vamos para a proxima
548                 }
549             }//END if(lido[0]=='i')
550             fputs(lido,arq_dst);                                    //Comentários a
                seguir
551             for(j=0;j<4;j++){                                       //Se o
                identificador buscado nao tiver correspondencia ou o ponteiro no
                "filmes.txt" nao tiver lido um identificador
552                 fgets(lido,55,arq_orig);                                //Leia a
                    linha do "filmes.txt"
553                 fputs(lido,arq_dst);                                    //Copie para
                    "filmes_copia.txt"
554             }
555         }//END else
556
557     }while(*tam!=0 || erro!=NULL);                                  //Faca a
        gravacao ate o tamanho ser zero, ou ate acabarmos de ler o banco de dados
558
559     fclose(arq_orig);                                               //Feche os
        ponteiros de arquivo por segurança
560     fclose(arq_dst);
561     remove("filmes.txt");                                           //Remove o
        antigo "filmes.txt"
562     rename("filmes_copia.txt","filmes.txt");                        //Renomeia a
        copia para "filmes.txt"
563     return 0;                                                       //Retorna 0
564 }
565 /*
566 Função: leFilmes
567 Autor: Feita por Gabriel Henrique
568 */
569 int leFilmes(Filme *ptr,int acao,FILE *dados){
570     limpa_tela();
571     char auxiliar[55],*teste_de_fim;
572     int quantidade_adicionada=0,posicao_identificador,n;
573     if(acao==2){
574         fseek(dados,0,SEEK_SET);
575         do
576         {
577             teste_de_fim=fgets(auxiliar,55,dados);
578             if(teste_de_fim!=NULL)puts(auxiliar);
579             if(auxiliar[0]=='i'){
580                 quantidade_adicionada++;
581                 if(quantidade_adicionada==1)posicao_identificador=ftell(dados)-strlen
                    (auxiliar);    //Vamos salvar a posição desse identificador no
                    arquivo, para que assim no futuro possamos salvar os filmes
582             }
583             if(teste_de_fim==NULL)printf("\nFIM DO BANCO DE DADOS");
584             if(quantidade_adicionada==50 || teste_de_fim==NULL){
585                 printf("\nDeseja salvar estes filmes?(Digite S para sim e N para
                    nao)\n");
586                 auxiliar[1]=getchar();
587                 limpa_buffer();
588                 auxiliar[1]=tolower(auxiliar[1]);
589                 if(auxiliar[1]=='n' && teste_de_fim!=NULL)quantidade_adicionada=0;
590             }
591         } while (auxiliar[1]!='s');
592         fseek(dados,posicao_identificador,SEEK_SET);
593         for(n=0;n<quantidade_adicionada;n++){
594             fscanf(dados,"%*s%d",&ptr[n].identificador );
595             fscanf(dados,"%*s %[^\n]",ptr[n].nome);         //Leia uma string até
                encontrar o \n e descarte o \n
596             fscanf(dados,"%*s %[^\n]",ptr[n].genero);       //Leia uma string até
                encontrar o \n e descarte o \n
597             fscanf(dados,"%*s %d",&ptr[n].anoLancamento);
598             fscanf(dados,"%*s %[^\n]",ptr[n].nomeDiretor);  //Leia uma string até
                encontrar o \n e descarte o \n
599         }
```

```
600         }
601         else{
602             imprimeFilmes(&(*ptr),-1);
603             printf("\n\n\nO FILME BUSCADO FOI ENCONTRADO\n");
604             printf("\nQual posicao deseja colocar o filme encontrado?\n(Se nenhum filme
                aparecer acima, digite um numero entre 1 e 50)\n");
605             le_numero(&n,1,50,'i');
606             n--;
607             fscanf(dados,"%*s%d",&ptr[n].identificador );
608             fscanf(dados,"%*s %[^\n]",ptr[n].nome);         //Leia uma string até
                encontrar o \n e descarte o \n
609             fscanf(dados,"%*s %[^\n]",ptr[n].genero);       //Leia uma string até
                encontrar o \n e descarte o \n
610             fscanf(dados,"%*s %d",&ptr[n].anoLancamento);
611             fscanf(dados,"%*s %[^\n]",ptr[n].nomeDiretor);  //Leia uma string até
                encontrar o \n e descarte o \n
612         }
613         return quantidade_adicionada;
614     }
615     /*
616     Função: buscaFilme
617     Autor: Feita por Henrique Soares Costa, github.com/RIQUETRET
618     Entradas: O Vetor de filmes(*ptr), O ponteiro de arquivos(*ptr2)
619     e por fim o nome do filme desejado (*nome_filme)
620     Saídas: Seu vetor recebe o filme desejado, se este filme
621     for encontrado na base de dados
622     Retorno: Zero se tudo ok ou -1 se nao encontrou o filme
623
624     Objetivo: Vasculha o arquivo, lendo linha por linha
625     quando encontra um nome de filme igual ao desejado,
626     carrega no vetor com leFilmes() e posteriomente retorna 0
627     */
628
629     int buscaFilme(Filme *ptr,FILE *ptr2,char *nome_filme){
630         int posicao_identificador; //Variavel para armazenar a posicao do "i" ou
            identificador, para que assim possamos reposicionar o ponteiro de arquivos (*ptr2)
631         char lido[55];  //Vamos ler linha por linha do nosso arquivo com esse vetor e
            portanto podemos ter ao máximo 54 caracteres, sendo que isso pode ocorrer quando
            formos ler o nome, ou seja, "n: \n" + 50 caracteres do nome
632
633         fseek(ptr2,0,SEEK_SET);    //Posiciona o cursor no inicio para buscar o filme
634
635         while(fgets(lido,55,ptr2)!=NULL) {                          //Leia uma linha do
            arquivo e caso encontre EOF, saia do loop e retorne -1
636             if (lido[0]=='i'){                                      //A gente leu um
                identificador?
637                 posicao_identificador=ftell(ptr2)-strlen(lido);     //Vamos salvar a
                    posição desse identificador, para que assim no futuro possamos chamar a
                    função le_filmes, que no caso precisa obrigatoriamente iniciar a leitura
                    no identificador
638             }
639             else if(lido[0]=='n'){                                  //A gente leu um
                filme?
640                 lido[strcspn(lido, "\n")] = 0;                      //Remove \n
                    introduzido pelo fgets
641                 if(strcmp(&lido[3],nome_filme)==0){                 //O filme lido e o
                    desejado eh igual?
642                     fseek(ptr2,posicao_identificador,SEEK_SET);     //Reposiciona cursor
                        para o identificador deste filme
643                     leFilmes(&(*ptr),1,ptr2);                       //Le o filme (salva
                        para o vetor de filmes)
644                     return 0;                                       //Retorna 0
                        indicando sucesso na busca e salvamento do filme
645                 }
646             }
647         }//END while(fgets(lido,55,*ptr2)!=NULL)
648         return -1;                                                  //Retorna -1
            indicando falha na busca
649     }
```