Visão Geral

Este programa em C realiza a leitura de um arquivo .csv contendo dados de filmes, armazena essas informações em uma estrutura de dados (struct) e permite ordenar os filmes com base em diferentes critérios: nome, categoria, nota e duração, e pode exportar os dados processados, novamente para um arquivo .csv.

Estrutura do Código

Bibliotecas Utilizadas

- stdio.h: operações de entrada/saída.
- stdlib.h: funções utilitárias (alocação de memória, conversões, etc.).
- string.h: manipulação de strings.

Estruturas Utilizadas

- **nome**: Nome do filme.
- **categoria**: Categoria/ gênero do filme.
- **nota**: Nota de avaliação do filme.
- **duração**: Duração do filme (em horas e minutos).

Definições

- MAX_FILMES: quantidade máxima de filmes (100).
- TAM_NOME_FILME: tamanho máximo do nome do filme (30 caracteres).
- TAM_NOME_CATEGORIA: tamanho máximo do nome da categoria (12 caracteres).

Funções Principais

int charToAscii(char c)

- Converte um caractere em seu valor ASCII.
- Auxilia na comparação entre caracteres nas funções de ordenação.

int recuperarConteudoExcel(Filme filmes[])

- Lê os dados do arquivo dados.csv.
- Armazena cada linha em um elemento do vetor de Filme.
- Ignora a primeira linha do arquivo (cabeçalho).

- Retorno: quantidade de filmes lidos.

void mostrarDadosOrdenados(Filme filmes[], int qtd)

- Exibe na tela as informações de todos os filmes cadastrados.
- Formata a duração no padrão "XhYmin".

Funções de Ordenação

Todas as funções de ordenação implementam o algoritmo Bubble Sort, personalizado para lidar com a estrutura Filme, tal função também foi feita para analisar o nome do filme letra a letra. Durante a ordenação, sempre que dois filmes trocam de posição, todos os campos (nome, categoria, nota, duração) são trocados simultaneamente, tornando a função cada vez mais a prova de falhas.

void ordenarFilmesPorNome(Filme filmes[], int qtd)

- Ordena os filmes em ordem alfabética com base no nome do filme.
- Comparação é feita caractere por caractere usando valores ASCII.

void ordenarFilmesPorCategoria(Filme filmes[], int qtd)

- Ordena os filmes em ordem alfabética com base na categoria (gênero).

void ordenarFilmesPorAvaliacao(Filme filmes[], int qtd)

- Ordena os filmes da nota mais baixa para a mais alta.

void ordenarFilmesPorDuracao(Filme filmes[], int qtd)

- Ordena os filmes do mais curto para o mais longo, considerando a duração em minutos.

Arquivo de Entrada - dados.csv

- Formato esperado:

  Nome, Categoria, Nota, Duração

  Filme A, Aventura, 8.5,120

  Filme B, Drama, 7.2, 90

  Filme C, Comédia, 6.8,110

- Cada linha representa um filme.

- As colunas são separadas por vírgula.

Problemas Encontrados

```
void ordenarFilmesPorNome(Filme filmes[], int qtd)
{

    char aux[TAM_NOME_FILME];
    float apoio = 0;
    for (int i = 0; i < qtd - 1; i++)
    {
        for (int k = 0; k < TAM_NOME_FILME; k++)
        {
            if (charToAscii(filmes[i].nome[k]) > charToAscii(filmes[i + 1].nome[k]) || filmes[i + 1].nome[k] == '\n')
            {
                strcpy(aux, filmes[i + 1].nome);
                strcpy(filmes[i + 1].nome, filmes[i].nome);
                strcpy(filmes[i].nome, aux);

                strcpy(aux, filmes[i + 1].categoria);
                strcpy(filmes[i + 1].categoria, filmes[i].categoria);
                strcpy(filmes[i].categoria, aux);

                apoio = filmes[i + 1].nota;
                filmes[i + 1].nota = filmes[i].nota;
                filmes[i].nota = apoio;

                apoio = filmes[i + 1].duracao;
                filmes[i + 1].duracao = filmes[i].duracao;
                filmes[i].duracao = apoio;
                i = -1;
                break;
            }
            else if (filmes[i].nome[k] == '\n')
            {
                break;
            }
            else if (charToAscii(filmes[i].nome[k]) == charToAscii(filmes[i + 1].nome[k]))
            {
                continue;
            }
```

O nosso código teve de ser modificado posterior a um erro encontrado, quando executado, não havia precisão na ordenação por números, o que foi modificado na nova versão como mostra o código a seguir.

```
char aux[TAM_NOME_FILME];
int i, j;
float apoio = 0;
for (i = 0; i < qtd - 1; i++) {
    printf("\n[%d] ", i);

    for (j = 0; j < qtd - i - 1; j++) {
        printf("%d, ", j);

        for (int k = 0; k < TAM_NOME_FILME; k++)
        {
            if (charToAscii(filmes[j].nome[k]) > charToAscii(filmes[j + 1].nome[k]) || filmes[j + 1].nome[k] == '\n')
            {
                strcpy(aux, filmes[j + 1].nome);
                strcpy(filmes[j + 1].nome, filmes[j].nome);
                strcpy(filmes[j].nome, aux);

                strcpy(aux, filmes[j + 1].categoria);
                strcpy(filmes[j + 1].categoria, filmes[j].categoria);
                strcpy(filmes[j].categoria, aux);

                apoio = filmes[j + 1].nota;
                filmes[j + 1].nota = filmes[j].nota;
                filmes[j].nota = apoio;

                apoio = filmes[j + 1].duracao;
                filmes[j + 1].duracao = filmes[j].duracao;
                filmes[j].duracao = apoio;
                break;
            }
            else if (filmes[j].nome[k] == '\n')
            {
                break;
            }
            else if (charToAscii(filmes[j].nome[k]) == charToAscii(filmes[j + 1].nome[k]))
            {
                continue;
```

Funcionamento

1. O programa vai ler os dados de filmes de um arquivo CSV.
2. O usuário escolhe o critério de ordenação.
3. Os filmes são ordenados conforme a escolha do usuário.
4. O resultado pode ser exibido e posteriormente exportado para um novo CSV.