Pontifícia Universidade Católica do Rio Grande do Sul – PUCRS
Escola Politécnica – Curso de Bacharelado em Engenharia de *Software*
Probramação de *Software* Básico – Turma 33 – Prof. Roland Teodorowitsch – 2025/1

TRABALHO 1: PROGRAMAÇÃO DE CINEMA

Introdução

O objetivo deste trabalho é praticar a programação utilizando a linguagem C. Este trabalho consiste em modelar e implementar um sistema para mostrar dados da programação de cinemas em determinada cidade em um dia específico, incluindo arquivos com a especificação de filmes, cinemas e exibições.

Definição

Este trabalho, que deverá ser feito **individualmente**, consiste em ler diversas informações armazenadas em arquivos de texto, processando estas informações e gerando um relatório com a programação da exibição de filmes nos cinemas de determinada cidade.

A sua aplicação deverá gerenciar informações sobre:

- Filmes (com informações sobre cada filme);
- Cinemas (com dados de cada cinema, incluindo endereço e número de salas);
- Programação (que indica qual filme que será exibido em determinada sala de cinema, incluindo horários das sessões).

Recomenda-se que cada uma destas entidades seja gerenciada por um conjunto padronizado de funções. Uma sugestão de funções a serem implementadas para cada entidade será apresentada a seguir.

Além das funções para gerenciamento dos dados das entidades, recomenda-se também a implementação de um conjunto básico de operações que a aplicação deve executar.

Filmes

Os dados de filmes serão armazenados em um arquivo de texto, com cada filme sendo representado por 4 linhas neste arquivo. Para cada filme, as suas respectivas 4 linhas conterão: o identificador do filme (número inteiro), o título ou nome do filme, a faixa etária e o estilo. A Figura 1 mostra um exemplo de arquivo de filmes neste formato.

```
1
Besouro Azul
14
Ação
2
As Tartarugas Ninja: Caos Mutante
10
Animação
3
Barbie
12
Comédia
4
O Porteiro
14
Comédia
5
Oppenheimer
16
Drama
6
```

```
Presos na Floresta - Fuja se For Capaz
14
Suspense
7
Drácula: a Última Viagem de Demeter
16
Terror
8
Golda: a Mulher de Uma Nação
14
Drama
9
Toc Toc Toc: Ecos do Além
16
Terror
10
Fale Comigo
16
Terror
11
Gran Turismo - de Jogador a Corredor
12
Ação
12
A Chamada
14
Ação
13
Asteroid City
14
Dramas
```

Figura 1 – Arquivo de filmes (`filmes.txt`)

Para armazenar as informações de filmes será necessário declarar um vetor de filmes. O vetor deverá ser capaz de armazenar até 100 filmes. Por sua vez, a estrutura (registro) para cada filme deverá ser composta por 4 campos: identificador (que identifica o filme na programação de cinema), título do filme (cadeia de caracteres com espaço para até 60 caracteres), faixa etária (valor inteiro, sendo que 0 indica classificação livre) e estilo (cadeia de caracteres com espaço para até 20 caracteres).

Para gerenciar as informações de filmes, recomenda-se a implementação das seguintes funções:

- `int filmes_menor(filme_t *f1, filme_t *f2)`: recebe duas referências para filmes e retorna 1 (verdadeiro) se o filme da primeira referência for menor do que o filme da segunda referência (usando o título como critério de ordenação), ou 0 (falso) em caso contrário;
- `int filmes_carrega(char *nome_arq)`: lê o arquivo cujo nome foi indicado, trazendo as informações do arquivo para o vetor de filmes – esta função retorna 0 (falso) em caso de erro ou 1 (verdadeiro) em caso de sucesso;
- `int filmes_num_filmes()`: retorna o número de filmes no vetor de filmes;
- `filme_t *filmes_obtem_filme_id(int id)`: retorna o endereço do filme com o identificador `id` no vetor de filmes, ou `NULL` caso não exista nenhum filme com este identificador;
- `filme_t *filme_obtem_filme_indice(int indice)`: retorna o endereço do filme com o `indice` especificado no vetor de filmes, ou `NULL` caso o índice seja inválido (menor do que 0 ou maior ou igual ao número de filmes no vetor);
- `void filmes_mostra_filme(filme_t *filme)`: mostra o filme apontado pelo ponteiro `filme`, que deverá ser exibido no seguinte formato: título, " [", identificador, "] - ", faixa etária seguida de " anos" (ou "LIVRE"), " - " e estilo;
- `void filmes_mostra()`: mostra todos os filmes do vetor de filmes, usando para isso a função `filmes_mostra_filme()`;
- `void filmes_ordena()`: ordena o vetor de filmes, usando a função `filmes_menor()` para comparar os filmes;
- `int filmes_salva(char *nome_arq)`: transfere os dados no vetor de filmes para o

arquivo cujo nome foi indicado – esta função retorna 0 (falso) em caso de erro ou 1 (verdadeiro) em caso de sucesso.

Para testar a implementação destas funções, pode-se usar como exemplo o programa principal mostrado na Figura 2.

```
// app1.cpp (Roland Teodorowitsch; 17 maio 2024)

#include <stdio.h>
#include "filmes.h"

int main() {
  if ( !filmes_carrega("filmes.txt") ) return 0;
  printf("----------\n");
  filmes_mostra();
  filmes_ordena();
  if ( !filmes_salva("filmes.txt.out") ) return 0;
  printf("----------\n");
  filmes_mostra();
  printf("----------\n");
  return 0;
}
```

Figura 2 – Arquivo `app1.c`

O programa `app1.c` lê o arquivo de filmes `filmes.txt`, mostra estes filmes, ordena o vetor de filmes (pelo título), mostra o vetor ordenado e salva uma versão do vetor de filmes no arquivo `filmes.txt.out`. A Figura 3 mostra a saída esperada para a execução de `app1.c`. E a Figura 4 mostra o conteúdo esperado para o arquivo `filmes.txt.out`.

```
----------
Besouro Azul [1] - 14 anos - Ação
As Tartarugas Ninja: Caos Mutante [2] - 10 anos - Animação
Barbie [3] - 12 anos - Comédia
O Porteiro [4] - 14 anos - Comédia
Oppenheimer [5] - 16 anos - Drama
Presos na Floresta - Fuja se For Capaz [6] - 14 anos - Suspense
Drácula: a Última Viagem de Demeter [7] - 16 anos - Terror
Golda: a Mulher de Uma Nação [8] - 14 anos - Drama
Toc Toc Toc: Ecos do Além [9] - 16 anos - Terror
Fale Comigo [10] - 16 anos - Terror
Gran Turismo - de Jogador a Corredor [11] - 12 anos - Ação
A Chamada [12] - 14 anos - Ação
Asteroid City [13] - 14 anos - Dramas
----------
A Chamada [12] - 14 anos - Ação
As Tartarugas Ninja: Caos Mutante [2] - 10 anos - Animação
Asteroid City [13] - 14 anos - Dramas
Barbie [3] - 12 anos - Comédia
Besouro Azul [1] - 14 anos - Ação
Drácula: a Última Viagem de Demeter [7] - 16 anos - Terror
Fale Comigo [10] - 16 anos - Terror
Golda: a Mulher de Uma Nação [8] - 14 anos - Drama
Gran Turismo - de Jogador a Corredor [11] - 12 anos - Ação
O Porteiro [4] - 14 anos - Comédia
Oppenheimer [5] - 16 anos - Drama
Presos na Floresta - Fuja se For Capaz [6] - 14 anos - Suspense
Toc Toc Toc: Ecos do Além [9] - 16 anos - Terror
----------
```

Figura 3 – Saída esperada para a execução de `app1.c`

```
12
A Chamada
14
Ação
2
As Tartarugas Ninja: Caos Mutante
10
Animação
13
Asteroid City
14
Dramas
3
Barbie
12
```

```
Comédia
1
Besouro Azul
14
Ação
7
Drácula: a Última Viagem de Demeter
16
Terror
10
Fale Comigo
16
Terror
8
Golda: a Mulher de Uma Nação
14
Drama
11
Gran Turismo - de Jogador a Corredor
12
Ação
4
O Porteiro
14
Comédia
5
Oppenheimer
16
Drama
6
Presos na Floresta - Fuja se For Capaz
14
Suspense
9
Toc Toc Toc: Ecos do Além
16
Terror
```

Figura 4 – Arquivo `filmes.txt.out` gerado a partir da execução de `app1.c`

Cinemas

Os dados de cinemas serão armazenados em um arquivo de texto, com cada cinema sendo representado por 10 linhas neste arquivo. Para cada cinema, as suas respectivas 10 linhas conterão: o identificador do cinema (número inteiro), o nome do cinema, logradouro, número (do logradouro – valor inteiro, com -1 indicando "s/n"), complemento (do logradouro), bairro, CEP, cidade, estado e número de salas. A Figura 5 mostra um exemplo de arquivo de cinemas neste formato.

```
1
GNC Praia de Belas
Rua Praia de Belas
1181
Loja 3038 - Terceiro Piso
Menino Deus
90110-000
Porto Alegre
RS
6
2
GNC Iguatemi
Avenida João Wallig
1800

Passo da Areia
91349-900
Porto Alegre
RS
6
3
GNC Moinhos
Rua Olavo Barreto Viana
36
4º Andar
Moinhos de Vento
90570-070
Porto Alegre
RS
4
```

```
4
GNC Lindóia
Avenida Assis Brasil
3522
Lojas 301/302
Jardim Lindóia
91010-007
Porto Alegre
RS
2
```

Figura 5 – Arquivo de cinemas (`cinemas.txt`)

Para armazenar as informações de cinemas será necessário declarar um vetor de cinemas. O vetor deverá ser capaz de armazenar até 50 cinemas. Por sua vez, a estrutura (registro) para cada cinema deverá ser composta por 10 campos: identificador (que identifica o cinema na programação de cinema), nome do cinema (cadeia de caracteres com espaço para até 60 caracteres), logradouro do cinema (cadeia de caracteres com espaço para até 60 caracteres), número do logradouro (valor inteiro, com -1 sendo usado para indicar logradouro sem número – "s/n"), complemento do logradouro (cadeia de caracteres com espaço para até 40 caracteres), bairro (cadeia de caracteres com espaço para até 40 caracteres), CEP (cadeia de caracteres com espaço para até 9 caracteres), cidade (cadeia de caracteres com espaço para até 40 caracteres), estado (cadeia de caracteres com espaço para até 2 caracteres) e número de salas (valor inteiro).

Para gerenciar as informações de cinemas, recomenda-se a implementação das seguintes funções:

- `int cinemas_menor(cinema_t *c1, cinema_t *c2)`: recebe duas referências para cinemas e retorna 1 (verdadeiro) se o cinema da primeira referência for menor do que o cinema da segunda referência (usando o nome como critério de ordenação), ou 0 (falso) em caso contrário;
- `int cinemas_carrega(char *nome_arq)`: lê o arquivo cujo nome foi indicado, trazendo as informações do arquivo para o vetor de cinemas – esta função retorna 0 (falso) em caso de erro ou 1 (verdadeiro) em caso de sucesso;
- `int cinemas_num_cinemas()`: retorna o número de cinemas no vetor de cinemas;
- `cinema_t *cinemas_obtem_cinema_id(int id)`: retorna o endereço do cinema com o identificador `id` no vetor de cinemas, ou `NULL` caso não exista nenhum cinema com este identificador;
- `cinema_t *cinemas_obtem_cinema_indice(int indice)`: retorna o endereço do cinema com o `indice` especificado no vetor de cinemas, ou `NULL` caso o índice seja inválido (menor do que 0 ou maior ou igual ao número de cinemas no vetor);
- `void cinemas_mostra_cinema(cinema_t *cinema)`: mostra o cinema apontado pelo ponteiro `cinema`, que deverá ser exibido no seguinte formato: nome, " [", número, "] ", nova linha, logradouro, ", ", número, " - ", complemento, " - Bairro ", bairro, " - CEP ", cep, " - ", cidade, " - ", estado, nova linha, "Cinema(s): ", número de salas e nova linha;
- `void cinemas_mostra()`: mostra todos os cinemas do vetor de cinemas, usando para isso a função `cinemas_mostra_cinema()`;
- `void cinemas_ordena()`: ordena o vetor de cinemas, usando a função `cinemas_menor()` para comparar os cinemas;
- `int cinemas_salva(char *nome_arq)`: transfere os dados no vetor de cinemas para o arquivo cujo nome foi indicado – esta função retorna 0 (falso) em caso de erro ou 1 (verdadeiro) em caso de sucesso.

Para testar a implementação destas funções, pode-se usar como exemplo o programa principal mostrado na Figura 6.

```c
// app2.c (Roland Teodorowitsch; 17 maio 2024)

#include <stdio.h>
#include "cinemas.h"

int main() {
  if ( !cinemas_carrega("cinemas.txt") ) return 0;
  printf("----------\n");
  cinemas_mostra();
  cinemas_ordena();
  if ( !cinemas_salva("cinemas.txt.out") ) return 0;
  printf("----------\n");
  cinemas_mostra();
  printf("----------\n");
  return 0;
}
```

Figura 6 – Arquivo `app2.c`

O programa `app2.c` lê o arquivo de cinemas `cinemas.txt`, mostra estes cinemas, ordena o vetor de cinemas (pelo título), mostra o vetor ordenado e salva uma versão do vetor de cinemas no arquivo `cinemas.txt.out`. A Figura 7 mostra a saída esperada para a execução de `app2.c`. E a Figura 8 mostra o conteúdo esperado para o arquivo `cinemas.txt.out`.

```
----------
GNC Praia de Belas [1]
Rua Praia de Belas, 1181 - Loja 3038 - Terceiro Piso - Bairro Menino Deus - CEP 90110-000 - Porto
Alegre - RS
Cinema(s): 6

GNC Iguatemi [2]
Avenida João Wallig, 1800 - Bairro Passo da Areia - CEP 91349-900 - Porto Alegre - RS
Cinema(s): 6

GNC Moinhos [3]
Rua Olavo Barreto Viana, 36 - 4º Andar - Bairro Moinhos de Vento - CEP 90570-070 - Porto Alegre - RS
Cinema(s): 4

GNC Lindóia [4]
Avenida Assis Brasil, 3522 - Lojas 301/302 - Bairro Jardim Lindóia - CEP 91010-007 - Porto Alegre - RS
Cinema(s): 2

----------
GNC Iguatemi [2]
Avenida João Wallig, 1800 - Bairro Passo da Areia - CEP 91349-900 - Porto Alegre - RS
Cinema(s): 6

GNC Lindóia [4]
Avenida Assis Brasil, 3522 - Lojas 301/302 - Bairro Jardim Lindóia - CEP 91010-007 - Porto Alegre - RS
Cinema(s): 2

GNC Moinhos [3]
Rua Olavo Barreto Viana, 36 - 4º Andar - Bairro Moinhos de Vento - CEP 90570-070 - Porto Alegre - RS
Cinema(s): 4

GNC Praia de Belas [1]
Rua Praia de Belas, 1181 - Loja 3038 - Terceiro Piso - Bairro Menino Deus - CEP 90110-000 - Porto
Alegre - RS
Cinema(s): 6

----------
```

Figura 7 – Saída esperada para a execução de `app2.c`

```
2
GNC Iguatemi
Avenida João Wallig
1800

Passo da Areia
91349-900
Porto Alegre
RS
6
4
GNC Lindóia
Avenida Assis Brasil
3522
Lojas 301/302
```

```
Jardim Lindóia
91010-007
Porto Alegre
RS
2
3
GNC Moinhos
Rua Olavo Barreto Viana
36
4º Andar
Moinhos de Vento
90570-070
Porto Alegre
RS
4
1
GNC Praia de Belas
Rua Praia de Belas
1181
Loja 3038 - Terceiro Piso
Menino Deus
90110-000
Porto Alegre
RS
6
```

Figura 8 – Arquivo `cinemas.txt.out` gerado a partir da execução de `app2.c`

Programação

Os dados com a programação dos filmes em exibição em cada sala de cinema serão armazenados em um arquivo de texto, com cada "programa" (item da programação) sendo representado por 5 linhas neste arquivo. Para cada programa, as suas respectivas 5 linhas conterão: identificador do cinema (número inteiro), sala do cinema (número inteiro), identificador do filme (número inteiro), tipo de exibição (número inteiro: 1 para legendado, 2 para dublado, 3 para 3D legendado, 4 para 3D dublado e 5 para Nacional) e horários de exibição (cadeia de caracteres). A Figura 9 mostra um exemplo de arquivo de cinemas neste formato.

```
1
1
1
2
14:10 16:45 19:15
1
1
1
1
21:45
1
2
2
4
14:25
1
2
2
2
16:30 18:40
1
2
2
1
21:00
1
3
3
2
16:10
1
3
4
5
14:00 18:50
1
3
5
```

```
1
20:45
1
4
6
2
20:00
1
4
7
2
13:20
1
4
6
1
15:40
1
4
7
1
17:40
1
4
8
1
22:00
1
5
9
2
13:45 21:50
1
5
9
1
17:50
1
5
10
2
19:50
1
5
10
1
15:50
1
6
8
1
16:20
1
6
11
2
19:00
1
6
11
1
13:30
1
6
3
1
21:40
2
1
5
1
20:45
2
1
3
2
16:10
2
1
4
5
14:10 18:50
2
2
```

```
8
1
17:40 22:00
2
2
4
5
15:35
2
2
2
1
20:00
2
2
7
1
13:15
2
3
10
1
19:45
2
3
9
2
13:50 17:50
2
3
9
1
21:50
2
3
10
2
15:50
2
4
1
2
13:30 18:40
2
4
1
1
16:00 21:25
2
5
2
1
21:10
2
5
2
4
14:20
2
5
2
2
16:40 19:00
2
6
11
1
21:40
2
6
3
1
19:15
2
6
11
2
13:40 16:20
3
1
12
1
14:45 21:20
3
```

```
1
13
1
17:00 19:15
3
2
5
1
14:00 17:20 20:45
3
3
3
1
14:15 16:30 18:45 21:00
3
4
8
1
14:30 16:45 19:00 21:30
```

Figura 9 – Arquivo de programação (`programacao.txt`)

Para armazenar as informações de programação será necessário declarar um vetor de programas. O vetor deverá ser capaz de armazenar até 100 programas. Por sua vez, a estrutura (registro) para cada programa deverá ser composta por 5 campos: referência para o cinema (ou seja ponteiro para o respectivo cinema no vetor de cinemas), sala do cinema (número inteiro), referência para o filme (ou seja, ponteiro para o respectivo filme no vetor de filmes), tipo de exibição (número inteiro: 1 para legendado, 2 para dublado, 3 para 3D legendado, 4 para 3D dublado e 5 para Nacional) e horários de exibição (cadeia de caracteres com espaço para 60 caracteres).

Para gerenciar as informações de cinemas, recomenda-se a implementação das seguintes funções:

- `int programacao_carrega(char *nome_arq)`: lê o arquivo cujo nome foi indicado, trazendo as informações do arquivo para o vetor de programação – esta função retorna 0 (falso) em caso de erro ou 1 (verdadeiro) em caso de sucesso – para o correto funcionamento desta função, deve-se executar antes as funções `filmes_carrega()` e `cinemas_carrega()`, assim será possível usar respectivamente as funções `filmes_obtem_filme_id()` e `cinemas_obtem_cinema_id()` para obter as referências para cada filme e cinema;
- `void programacao_mostra()`: esta função gera um relatório da programação – para cada cinema é mostrado o seu nome, seguido de toda programação para este cinema. A programação deve ser apresentada no seguinte formato: “Sala “ seguido do número da sala e de “: ”; nome do filme seguido de “ | ”; horários das sessões seguidos de “ | ”; tipo de exibição (usando os seguintes rótulos “LEG”, “DUB”, “3D LEG”, “3D DUB” e “NAC”, respectivamente, para 1, 2, 3, 4 e 5); faixa etária entre colchetes (ou “[LIVRE]” para faixa etária menor do que 1) seguida de “ | ” e estilo do filme.

Para testar a implementação destas funções, pode-se usar como exemplo o programa principal mostrado na Figura 10.

```c
// app3.c (Roland Teodorowitsch; 18 maio 2024)

#include <stdio.h>
#include "programacao.h"

int main() {
  if ( !filmes_carrega("filmes.txt") ) return 0;
  if ( !cinemas_carrega("cinemas.txt") ) return 0;
  cinemas_ordena();
  if ( !programacao_carrega("programacao.txt") ) return 0;
  printf("----------\n");
  programacao_mostra();
  printf("----------\n");
  return 0;
}
```

Figura 10 – Arquivo `app3.c`

Inicialmente programa `app3.c` lê o arquivo de filmes `filmes.txt`, lê o arquivo de cinemas `cinemas.txt`, ordena os cinemas usando a função `cinemas_ordena()` e lê o arquivo de programação `programacao.txt`. Por fim, este programa mostra a programação usando a função `programacao_mostra()`. A Figura 11 mostra a saída esperada para a execução de `app3.c`.

```
----------
GNC Iguatemi

Sala 1: Oppenheimer | 20:45 | LEG | [16] | Drama
Sala 1: Barbie | 16:10 | DUB | [12] | Comédia
Sala 1: O Porteiro | 14:10 18:50 | NAC | [14] | Comédia
Sala 2: Golda: a Mulher de Uma Nação | 17:40 22:00 | LEG | [14] | Drama
Sala 2: O Porteiro | 15:35 | NAC | [14] | Comédia
Sala 2: As Tartarugas Ninja: Caos Mutante | 20:00 | LEG | [10] | Animação
Sala 2: Drácula: a Última Viagem de Demeter | 13:15 | LEG | [16] | Terror
Sala 3: Fale Comigo | 19:45 | LEG | [16] | Terror
Sala 3: Toc Toc Toc: Ecos do Além | 13:50 17:50 | DUB | [16] | Terror
Sala 3: Toc Toc Toc: Ecos do Além | 21:50 | LEG | [16] | Terror
Sala 3: Fale Comigo | 15:50 | DUB | [16] | Terror
Sala 4: Besouro Azul | 13:30 18:40 | DUB | [14] | Ação
Sala 4: Besouro Azul | 16:00 21:25 | LEG | [14] | Ação
Sala 5: As Tartarugas Ninja: Caos Mutante | 21:10 | LEG | [10] | Animação
Sala 5: As Tartarugas Ninja: Caos Mutante | 14:20 | 3D DUB | [10] | Animação
Sala 5: As Tartarugas Ninja: Caos Mutante | 16:40 19:00 | DUB | [10] | Animação
Sala 6: Gran Turismo - de Jogador a Corredor | 21:40 | LEG | [12] | Ação
Sala 6: Barbie | 19:15 | LEG | [12] | Comédia
Sala 6: Gran Turismo - de Jogador a Corredor | 13:40 16:20 | DUB | [12] | Ação

GNC Lindóia


GNC Moinhos

Sala 1: A Chamada | 14:45 21:20 | LEG | [14] | Ação
Sala 1: Asteroid City | 17:00 19:15 | LEG | [14] | Dramas
Sala 2: Oppenheimer | 14:00 17:20 20:45 | LEG | [16] | Drama
Sala 3: Barbie | 14:15 16:30 18:45 21:00 | LEG | [12] | Comédia
Sala 4: Golda: a Mulher de Uma Nação | 14:30 16:45 19:00 21:30 | LEG | [14] | Drama

GNC Praia de Belas

Sala 1: Besouro Azul | 14:10 16:45 19:15 | DUB | [14] | Ação
Sala 1: Besouro Azul | 21:45 | LEG | [14] | Ação
Sala 2: As Tartarugas Ninja: Caos Mutante | 14:25 | 3D DUB | [10] | Animação
Sala 2: As Tartarugas Ninja: Caos Mutante | 16:30 18:40 | DUB | [10] | Animação
Sala 2: As Tartarugas Ninja: Caos Mutante | 21:00 | LEG | [10] | Animação
Sala 3: Barbie | 16:10 | DUB | [12] | Comédia
Sala 3: O Porteiro | 14:00 18:50 | NAC | [14] | Comédia
Sala 3: Oppenheimer | 20:45 | LEG | [16] | Drama
Sala 4: Presos na Floresta - Fuja se For Capaz | 20:00 | DUB | [14] | Suspense
Sala 4: Drácula: a Última Viagem de Demeter | 13:20 | DUB | [16] | Terror
Sala 4: Presos na Floresta - Fuja se For Capaz | 15:40 | LEG | [14] | Suspense
Sala 4: Drácula: a Última Viagem de Demeter | 17:40 | LEG | [16] | Terror
Sala 4: Golda: a Mulher de Uma Nação | 22:00 | LEG | [14] | Drama
Sala 5: Toc Toc Toc: Ecos do Além | 13:45 21:50 | DUB | [16] | Terror
Sala 5: Toc Toc Toc: Ecos do Além | 17:50 | LEG | [16] | Terror
Sala 5: Fale Comigo | 19:50 | DUB | [16] | Terror
Sala 5: Fale Comigo | 15:50 | LEG | [16] | Terror
Sala 6: Golda: a Mulher de Uma Nação | 16:20 | LEG | [14] | Drama
Sala 6: Gran Turismo - de Jogador a Corredor | 19:00 | DUB | [12] | Ação
Sala 6: Gran Turismo - de Jogador a Corredor | 13:30 | LEG | [12] | Ação
Sala 6: Barbie | 21:40 | LEG | [12] | Comédia

----------
```

Figura 11 – Saída esperada para a execução de `app3.c`

Avaliação

Os trabalhos serão avaliados numa escala de 0 (ZERO) até 10 (DEZ), levando em consideração as características descritas neste documento. Serão utilizados os seguintes pesos nesta avaliação:

- 80%: a aplicação executou corretamente sem erros, apresentando o comportamento esperado, conforme descrito neste documento;

- 10%: o aluno usou padrões de programação adequados (indentação, programação estruturada, nomes de variáveis significativos, comentários documentando o código, etc.), gerando uma aplicação padronizada capaz de compilar e executar em mais de uma plataforma sem erros;
- 10%: o projeto foi acompanhado de um `Makefile` que permite a compilação automatizada da aplicação.

Trabalhos com trechos copiados integralmente ou parcialmente serão avaliados com a nota mínima (ZERO).

Entrega

O trabalho deve ser desenvolvido **individualmente**.

A data de entrega do trabalho é **03 de junho de 2025, antes do horário da aula**.

Cada aluno deverá entregar, através do moodle, todos os arquivos de código-fonte, juntamente com um arquivo `Makefile`, necessários para compilar e executar o projeto, compactados no formato ZIP. **Atenção: NÃO utilizar outros formatos!** Deverá ser possível compilar e executar corretamente o programa no sistema operacional Linux. Não serão aceitos trabalhos que não compilem corretamente no compilador g++ (compilador C++ padrão no sistema operacional Linux). Cada aluno também deverá apresentar a execução de sua aplicação (o que poderá ser feito de forma oral para o professor ou por escrito). **Não serão avaliados trabalhos que não forem apresentados!**

Qualquer entrega feita fora das definições estabelecidas neste documento poderá implicar diminuição de nota. Em caso de cópia de trabalhos serão avaliados com a nota mínima (zero).