TP - Sistema da Locadora

1.0

Generated by Doxygen 1.9.8

Chapter 1

Sistema da Locadora PDS

1.1 Descrição do Projeto

O Sistema da Locadora PDS é uma aplicação desenvolvida como trabalho final da disciplina de Programação e Desenvolvimento de Software II (PDS II) do curso de Sistemas de Informação na UFMG. Este projeto implementa um sistema abrangente de aluguel de filmes, proporcionando funcionalidades como cadastro de filmes e clientes, processos de locação, devolução de filmes e gerenciamento do catálogo.

1.2 Equipe

- Victoria Flores
- Raquel Xavier
- Flávio Soriano
- Isadora Horta
- Ana Paula Theobald

1.3 Contexto do Projeto

O Sistema da Locadora PDS visa otimizar as operações de uma locadora de filmes, oferecendo soluções para desafios comuns encontrados no gerenciamento de catálogos, controle de estoque, administração de clientes e operações de locação.

1.4 Tecnologias Utilizadas

- **Linguagem de Programação:** C++
- **Compilação:** Makefile
- **Documentação:** Doxygen
- **Testes:** Biblioteca doctest
- **Controle de Versão:** Git e GitHub

1.5 Estrutura do Projeto

O projeto é organizado em classes, cada uma responsável por aspectos específicos do sistema:

- `formatacao`: Implementa métodos para formatação de strings.
- `locacao`: Contém métodos e atributos relacionados às operações de locação.
- `filme`: Classe base que engloba atributos e métodos comuns a todos os tipos de filmes.
- `dvd`, `bluray`, `fita`: Classes que herdam de `filme` e implementam características específicas.
- `cliente`: Classe que gerencia informações e operações relacionadas aos clientes.
- `sistema`: Classe que administra operações do sistema, evitando sobrecarregar o arquivo principal.

1.6 Funcionalidades Principais

1. **Cadastro de Filmes e Clientes:**
   - Permite o registro de novos filmes e clientes no sistema.
2. **Locação e Devolução de Filmes:**
   - Facilita o processo de aluguel e devolução de filmes, mantendo o controle do estoque.
3. **Listagem do Catálogo e Clientes:**
   - Apresenta informações detalhadas sobre os filmes disponíveis e a base de clientes.

1.7 Aprendizados

Este projeto foi uma oportunidade valiosa para aprimorar nossas habilidades em programação, colaboração e gerenciamento de projetos. Os principais aprendizados incluem:

- Programação orientada a objetos em C++.
- Colaboração efetiva em um projeto conjunto.
- Uso avançado de Git e GitHub para controle de versão.
- Práticas de programação defensiva e tratamento de exceções.
- Criação e manutenção de Makefile.
- Geração de documentação utilizando Doxygen.

1.8 Github da Equipe

- **Victoria Flores**
- **Raquel Xavier**
- **Flávio Soriano**
- **Isadora Horta**
- **Ana Paula Theobald**

Chapter 2

pds2-projeto-final

CRC CARDS

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

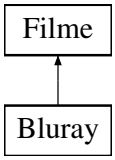
Chapter 6

Class Documentation

6.1 Bluray Class Reference

`#include <bluray.hpp>`

Inheritance diagram for Bluray:



Public Member Functions

- Bluray (const std::string &titulo, std::string codigo, int unidadesDisponiveis)

  Construtor da classe Bluray.
- void alugarAparelho ()

  Realiza o aluguel de um aparelho Blu-ray junto com o filme.
- void devolverAparelho ()

  Devolve o aparelho Blu-ray alugado.
- double calcularValorLocacao (int dias) const override

  Calcula o valor total da locação do Blu-ray.

Public Member Functions inherited from Filme

- Filme (const std::string &titulo, std::string codigo, Tipo tipo, int unidadesDisponiveis)

  Construtor da classe Filme.
- virtual ~Filme ()

  Destrutor da classe Filme (virtual para permitir polimorfismo).
- std::string getCodigo () const

  Getter para o código do filme.
- const std::string & getTitulo () const

  Getter para o título do filme.
- int getUnidadesDisponiveis () const

Getter para o número de unidades disponíveis.

- Tipo getTipo () const

  Getter para o tipo do filme.

- void adicionarUnidadesDisponiveis (int quantidade)

  Adiciona unidades disponíveis para locação.

- void removerUnidadeDisponivel ()

  Remove uma unidade disponível.

Static Public Member Functions

- static int getQuantidadeAparelhosDisponiveis ()

  Obtém a quantidade de aparelhos Blu-ray disponíveis para locação.

- static void setQuantidadeAparelhos (int quantidade)

  Define a quantidade de aparelhos Blu-ray disponíveis para locação.

Additional Inherited Members

Public Types inherited from Filme

- enum class Tipo { DVD , BluRay , Fita }

  Enumeração que representa os tipos de filme.

6.1.1 Detailed Description

A classe Bluray contém atributos como título, código, quantidade de unidades disponíveis, quantidade de aparelhos Blu-ray e informações sobre o aluguel de aparelhos.

Parameters

quantidadeAparelhos	Quantidade de aparelhos Blu-ray disponíveis para aluguel.
playerAlugado	Indica se um player Blu-ray foi alugado junto com o filme.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 Bluray()

```
Bluray::Bluray (
            const std::string & _titulo,
            std::string _codigo,
            int _unidadesDisponiveis )
```

Construtor da classe Bluray.

Parameters

titulo	Título do filme.
codigo	Código único do filme.

Parameters

unidadesDisponiveis	Número de unidades disponíveis para locação.
_titulo	Título do Blu-ray.
_codigo	Código único do Blu-ray.
_unidadesDisponiveis	Quantidade de unidades disponíveis do Blu-ray.

6.1.3 Member Function Documentation

6.1.3.1 alugarAparelho()

```
void Bluray::alugarAparelho ( )
```

Realiza o aluguel de um aparelho Blu-ray junto com o filme.

Realiza o aluguel de um aparelho de Blu-ray, se disponível.

Note

A quantidade de aparelhos disponíveis será reduzida.

Exceptions

std::invalid_argument	Se não houver aparelhos disponíveis.
std::invalid_argument	Se todos os aparelhos de Blu-ray estiverem alugados.

6.1.3.2 calcularValorLocacao()

```
double Bluray::calcularValorLocacao (
            int dias ) const  [override], [virtual]
```

Calcula o valor total da locação do Blu-ray.

Parameters

dias	Número de dias pelos quais o filme será alugado.

Returns

O valor total da locação.

Note

Adiciona um custo fixo se um aparelho Blu-ray foi alugado.

Parameters

dias	Número de dias pelos quais o filme será alugado.

Returns

O valor total da locação.

Implements Filme.

6.1.3.3 devolverAparelho()

```
void Bluray::devolverAparelho ( )
```

Devolve o aparelho Blu-ray alugado.

Devolve um aparelho de Blu-ray.

Note

A quantidade de aparelhos disponíveis será incrementada.

6.1.3.4 getQuantidadeAparelhosDisponiveis()

```
int Bluray::getQuantidadeAparelhosDisponiveis ( )  [static]
```

Obtém a quantidade de aparelhos Blu-ray disponíveis para locação.

Obtém a quantidade de aparelhos de Blu-ray disponíveis.

Returns

A quantidade de aparelhos disponíveis.

A quantidade de aparelhos de Blu-ray disponíveis.

6.1.3.5 setQuantidadeAparelhos()

```
void Bluray::setQuantidadeAparelhos (
            int quantidade )  [static]
```

Define a quantidade de aparelhos Blu-ray disponíveis para locação.

Define a quantidade total de aparelhos de Blu-ray.

Parameters

quantidade	Nova quantidade de aparelhos disponíveis.
quantidade	A nova quantidade total de aparelhos de Blu-ray.

The documentation for this class was generated from the following files:

- include/bluray.hpp
- src/Filme/bluray.cpp

6.2 Cliente Class Reference

`#include <cliente.hpp>`

Public Member Functions

- Cliente (string nome, string cpf)

  Construtor da classe Cliente.
- ~Cliente ()

  Destrutor da classe Cliente.
- void mostrarInfo ()

  Exibe informações do cliente.
- void adicionarPontos (int pontos)

  Adiciona pontos de fidelidade ao cliente.
- void usarPontos ()

  Usa os pontos de fidelidade do cliente.
- string getNome ()

  Obtém o nome do cliente.
- bool getAparelhoAlugado ()

  Obtém o status de alugel de aparelho do cliente.
- string getCpf ()

  Obtém o CPF do cliente.
- vector< Filme * > & getFilmesAlugados ()

  Obtém a lista de filmes alugados pelo cliente.
- int getPontos ()

  Obtém os pontos de fidelidade do cliente.
- void adicionarFilmeAlugado (Filme *filme)

  Adiciona um filme à lista de filmes alugados pelo cliente.
- void devolverFilmesAlugados ()

  Devolve todos os filmes alugados pelo cliente.
- void alugarAparelhoBluray ()

  Aluga o aparelho solicitado pelo cliente.
- void devolverAparelhoBluray ()

  Devolve o aparelho alugado pelo cliente.

6.2.1 Detailed Description

A classe Cliente possui atributos como nome, CPF, pontos de fidelidade e uma lista de filmes alugados. Além disso, contém métodos para exibir informações do cliente, adicionar pontos de fidelidade, zerar pontos, obter nome, CPF, lista de filmes alugados e pontos de fidelidade, adicionar filme à lista de filmes alugados e devolver todos os filmes alugados.

Parameters

_nome	Nome do cliente.
_cpf	CPF do cliente.
_pontos_fidelidade	Pontos de fidelidade acumulados pelo cliente.
_filmes_alugados	Lista de filmes alugados pelo cliente.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 Cliente()

```
Cliente::Cliente (
            string nome,
            string cpf )
```

Construtor da classe Cliente.

Parameters

nome	Nome do cliente.
cpf	CPF do cliente.

6.2.2.2 ~Cliente()

```
Cliente::~Cliente ( )
```

Destrutor da classe Cliente.

6.2.3 Member Function Documentation

6.2.3.1 adicionarFilmeAlugado()

```
void Cliente::adicionarFilmeAlugado (
            Filme * filme )
```

Adiciona um filme à lista de filmes alugados pelo cliente.

Parameters

filme	Filme a ser adicionado.
filme	Ponteiro para o filme a ser adicionado.

6.2.3.2 adicionarPontos()

```
void Cliente::adicionarPontos (
```

```
int pontos )
```

Adiciona pontos de fidelidade ao cliente.

Parameters

pontos	Pontos a serem adicionados.
pontos	Quantidade de pontos a serem adicionados.

6.2.3.3 alugarAparelhoBluray()

```
void Cliente::alugarAparelhoBluray ( )
```

Aluga o aparelho solicitado pelo cliente.

6.2.3.4 devolverAparelhoBluray()

```
void Cliente::devolverAparelhoBluray ( )
```

Devolve o aparelho alugado pelo cliente.

6.2.3.5 devolverFilmesAlugados()

```
void Cliente::devolverFilmesAlugados ( )
```

Devolve todos os filmes alugados pelo cliente.

6.2.3.6 getAparelhoAlugado()

```
bool Cliente::getAparelhoAlugado ( )
```

Obtém o status de alugel de aparelho do cliente.

Verifica se o cliente tem algum aparelho alugado em seu nome.

Returns

Bool do status.

Bool que representa o aluguel.

6.2.3.7 getCpf()

```
string Cliente::getCpf ( )
```

Obtém o CPF do cliente.

Returns

CPF do cliente.

6.2.3.8 getFilmesAlugados()

```
vector< Filme * > & Cliente::getFilmesAlugados ( )
```

Obtém a lista de filmes alugados pelo cliente.

Returns

Lista de filmes alugados.

Vetor contendo os filmes alugados.

6.2.3.9 getNome()

```
string Cliente::getNome ( )
```

Obtém o nome do cliente.

Returns

Nome do cliente.

6.2.3.10 getPontos()

```
int Cliente::getPontos ( )
```

Obtém os pontos de fidelidade do cliente.

Obtém a quantidade de pontos de fidelidade do cliente.

Returns

Pontos de fidelidade.

Quantidade de pontos de fidelidade.

6.2.3.11 mostrarInfo()

```
void Cliente::mostrarInfo ( )
```

Exibe informações do cliente.

Exibe as informações do cliente, incluindo nome, CPF e pontos de fidelidade.

6.2.3.12 usarPontos()

```
void Cliente::usarPontos ( )
```

Usa os pontos de fidelidade do cliente.

Utiliza os pontos de fidelidade do cliente.

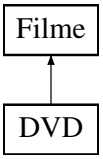
The documentation for this class was generated from the following files:

- include/cliente.hpp
- src/Cliente/cliente.cpp

6.3 DVD Class Reference

```
#include <dvd.hpp>
```

Inheritance diagram for DVD:



Public Member Functions

- DVD (const std::string &titulo, std::string codigo, int unidadesDisponiveis, Categoria _categoria)
  Construtor da classe DVD.
- double calcularValorLocacao (int dias) const override
  Calcula o valor total da locação do DVD.

Public Member Functions inherited from Filme

- Filme (const std::string &titulo, std::string codigo, Tipo tipo, int unidadesDisponiveis)
  Construtor da classe Filme.
- virtual ~Filme ()
  Destrutor da classe Filme (virtual para permitir polimorfismo).
- std::string getCodigo () const
  Getter para o código do filme.
- const std::string & getTitulo () const
  Getter para o título do filme.
- int getUnidadesDisponiveis () const
  Getter para o número de unidades disponíveis.
- Tipo getTipo () const
  Getter para o tipo do filme.
- void adicionarUnidadesDisponiveis (int quantidade)
  Adiciona unidades disponíveis para locação.
- void removerUnidadeDisponivel ()
  Remove uma unidade disponível.

Protected Attributes

- Categoria categoria

Additional Inherited Members

Public Types inherited from Filme

- enum class Tipo { DVD , BluRay , Fita }

  Enumeração que representa os tipos de filme.

6.3.1 Detailed Description

A classe DVD contém atributos como título, código, quantidade de unidades disponíveis e informações sobre a categoria do filme.

Parameters

categoria	Categoria do DVD (Lançamento, Estoque, Promoção).

6.3.2 Constructor & Destructor Documentation

6.3.2.1 DVD()

```
DVD::DVD (
            const std::string & _titulo,
            std::string _codigo,
            int _unidadesDisponiveis,
            Categoria _categoria )
```

Construtor da classe DVD.

Parameters

titulo	Título do filme.
codigo	Código único do filme.
unidadesDisponiveis	Número de unidades disponíveis para locação.
_categoria	Categoria do DVD (Lançamento, Estoque, Promoção).
_titulo	Título do DVD.
_codigo	Código único do DVD.
_unidadesDisponiveis	Quantidade de unidades disponíveis do DVD.
_categoria	Categoria do DVD (Lançamento, Estoque ou Promoção).

6.3.3 Member Function Documentation

6.3.3.1 calcularValorLocacao()

```
double DVD::calcularValorLocacao (
            int dias ) const  [override], [virtual]
```

Calcula o valor total da locação do DVD.

Calcula o valor de locação do DVD, considerando a categoria e a quantidade de dias.

Parameters

dias	Número de dias pelos quais o filme será alugado.

Returns

O valor total da locação.

Note

Adiciona um custo fixo baseado na categoria do DVD.

Parameters

dias	O número de dias para locação.

Returns

O valor total da locação do DVD.

Implements Filme.

6.3.4 Member Data Documentation

6.3.4.1 categoria

```
Categoria DVD::categoria  [protected]
```

Categoria do DVD (Lançamento, Estoque, Promoção).

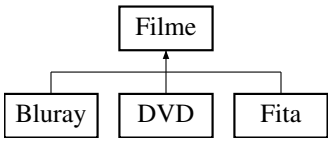
The documentation for this class was generated from the following files:

- include/dvd.hpp
- src/Filme/dvd.cpp

6.4 Filme Class Reference

`#include <filme.hpp>`

Inheritance diagram for Filme:



Public Types

- enum class Tipo { DVD , BluRay , Fita }

  Enumeração que representa os tipos de filme.

Public Member Functions

- Filme (const std::string &titulo, std::string codigo, Tipo tipo, int unidadesDisponiveis)

  Construtor da classe Filme.
- virtual ~Filme ()

  Destrutor da classe Filme (virtual para permitir polimorfismo).
- std::string getCodigo () const

  Getter para o código do filme.
- const std::string & getTitulo () const

  Getter para o título do filme.
- int getUnidadesDisponiveis () const

  Getter para o número de unidades disponíveis.
- Tipo getTipo () const

  Getter para o tipo do filme.
- void adicionarUnidadesDisponiveis (int quantidade)

  Adiciona unidades disponíveis para locação.
- void removerUnidadeDisponivel ()

  Remove uma unidade disponível.
- virtual double calcularValorLocacao (int dias) const =0

  Calcula o valor total da locação do filme.

Friends

- std::ostream & operator<< (std::ostream &os, const Filme::Tipo &tipo)

  Sobrecarga do operador de saída para imprimir o tipo do filme.

6.4.1 Detailed Description

A classe Filme contém atributos como título, código, quantidade de unidades disponíveis, tipo do filme (DVD, Blu↩ Ray, Fita) e métodos para manipulação desses atributos.

Parameters

titulo	Título do filme.
codigo	Código único do filme.
unidadesDisponiveis	Número de unidades disponíveis para locação.
tipo	Tipo do filme (DVD, BluRay, Fita).

6.4.2 Member Enumeration Documentation

6.4.2.1 Tipo

```
enum class Filme::Tipo  [strong]
```

Enumeração que representa os tipos de filme.

Enumerator

DVD	Tipo de filme: DVD.
BluRay	Tipo de filme: BluRay.
Fita	Tipo de filme: Fita.

6.4.3 Constructor & Destructor Documentation

6.4.3.1 Filme()

```
Filme::Filme (
            const std::string & titulo,
            std::string codigo,
            Filme::Tipo tipo,
            int unidadesDisponiveis )
```

Construtor da classe Filme.

Parameters

titulo	Título do filme.
codigo	Código único do filme.
tipo	Tipo do filme (DVD, BluRay, Fita).
unidadesDisponiveis	Número de unidades disponíveis para locação.
titulo	Título do filme.
codigo	Código único do filme.
tipo	Tipo do filme (DVD, BluRay, etc.).
unidadesDisponiveis	Quantidade de unidades disponíveis do filme.

6.4.3.2 ~Filme()

```
Filme::~Filme ( )  [virtual]
```

Destrutor da classe Filme (virtual para permitir polimorfismo).

Destrutor da classe Filme.

6.4.4 Member Function Documentation

6.4.4.1 adicionarUnidadesDisponiveis()

```
void Filme::adicionarUnidadesDisponiveis (
            int quantidade )
```

Adiciona unidades disponíveis para locação.

Adiciona unidades disponíveis ao filme.

Parameters

quantidade	A quantidade de unidades a serem adicionadas.

6.4.4.2 calcularValorLocacao()

```
double Filme::calcularValorLocacao (
            int dias ) const  [pure virtual]
```

Calcula o valor total da locação do filme.

Calcula o valor de locação do filme, considerando a quantidade de dias.

Parameters

dias	Número de dias pelos quais o filme será alugado.

Returns

O valor total da locação.

Note

Este método é virtual puro para permitir polimorfismo nas classes derivadas.

Parameters

dias	Número de dias para locação.

Returns

Valor total da locação do filme.

Implemented in Bluray, DVD, and Fita.

6.4.4.3 getCodigo()

```
std::string Filme::getCodigo ( ) const
```

Getter para o código do filme.

Obtém o código único do filme.

Returns

O código único do filme.

6.4.4.4 getTipo()

```
Filme::Tipo Filme::getTipo ( ) const
```

Getter para o tipo do filme.

Obtém o tipo do filme.

Returns

O tipo do filme (DVD, BluRay, Fita).

O tipo do filme (DVD, BluRay, etc.).

6.4.4.5 getTitulo()

```
const std::string & Filme::getTitulo ( ) const
```

Getter para o título do filme.

Obtém o título do filme.

Returns

O título do filme.

6.4.4.6 getUnidadesDisponiveis()

```
int Filme::getUnidadesDisponiveis ( ) const
```

Getter para o número de unidades disponíveis.

Obtém a quantidade de unidades disponíveis do filme.

Returns

O número de unidades disponíveis para locação.

A quantidade de unidades disponíveis do filme.

6.4.4.7 removerUnidadeDisponivel()

```
void Filme::removerUnidadeDisponivel ( )
```

Remove uma unidade disponível.

Remove uma unidade disponível do filme.

6.4.5 Friends And Related Symbol Documentation

6.4.5.1 operator<<

```
std::ostream & operator<< (
            std::ostream & os,
            const Filme::Tipo & tipo )  [friend]
```

Sobrecarga do operador de saída para imprimir o tipo do filme.

Parameters

os	Objeto de fluxo de saída.
tipo	Tipo do filme (DVD, BluRay, Fita).

Returns

O objeto de fluxo de saída.

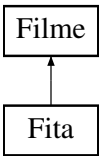
The documentation for this class was generated from the following files:

- include/filme.hpp
- src/Filme/filme.cpp

6.5 Fita Class Reference

`#include <fita.hpp>`

Inheritance diagram for Fita:



Public Member Functions

- Fita (const std::string &titulo, std::string codigo, int quantidade, bool _rebobinada)
  Construtor da classe Fita.
- bool estaRebobinada () const
  Verifica se a fita está rebobinada.
- void rebobinar ()
  Rebobina a fita.
- double calcularValorLocacao (int dias) const override
  Calcula o valor total da locação da fita.

Public Member Functions inherited from Filme

- Filme (const std::string &titulo, std::string codigo, Tipo tipo, int unidadesDisponiveis)
  Construtor da classe Filme.
- virtual ~Filme ()
  Destrutor da classe Filme (virtual para permitir polimorfismo).
- std::string getCodigo () const
  Getter para o código do filme.
- const std::string & getTitulo () const
  Getter para o título do filme.
- int getUnidadesDisponiveis () const
  Getter para o número de unidades disponíveis.
- Tipo getTipo () const
  Getter para o tipo do filme.
- void adicionarUnidadesDisponiveis (int quantidade)
  Adiciona unidades disponíveis para locação.
- void removerUnidadeDisponivel ()
  Remove uma unidade disponível.

Additional Inherited Members

Public Types inherited from Filme

- enum class Tipo { DVD , BluRay , Fita }
  Enumeração que representa os tipos de filme.

6.5.1 Detailed Description

A classe Fita contém atributos como título, código, quantidade de unidades disponíveis, informações sobre se a fita está rebobinada e métodos para manipulação desses atributos.

Parameters

rebobinada	Indica se a fita está rebobinada (true) ou não (false).

6.5.2 Constructor & Destructor Documentation

6.5.2.1 Fita()

```
Fita::Fita (
            const std::string & _titulo,
            std::string _codigo,
            int _quantidade,
            bool _rebobinada )
```

Construtor da classe Fita.

Parameters

titulo	Título do filme.
codigo	Código único do filme.
quantidade	Número de unidades disponíveis para locação.
_rebobinada	Indica se a fita está rebobinada (true) ou não (false).
_titulo	Título da fita.
_codigo	Código único da fita.
_quantidade	Quantidade de unidades disponíveis da fita.
_rebobinada	Estado da rebobinação da fita.

6.5.3 Member Function Documentation

6.5.3.1 calcularValorLocacao()

```
double Fita::calcularValorLocacao (
            int dias ) const  [override], [virtual]
```

Calcula o valor total da locação da fita.

Calcula o valor de locação da fita com base na quantidade de dias e no estado de rebobinação.

Parameters

dias	Número de dias pelos quais o filme será alugado.

Returns

O valor total da locação.

Parameters

dias	Quantidade de dias para locação.

Returns

O valor total da locação da fita.

Implements Filme.

6.5.3.2 estaRebobinada()

```
bool Fita::estaRebobinada ( ) const
```

Verifica se a fita está rebobinada.

Returns

True se a fita estiver rebobinada, false caso contrário.

Verdadeiro se a fita estiver rebobinada, falso caso contrário.

6.5.3.3 rebobinar()

```
void Fita::rebobinar ( )
```

Rebobina a fita.

The documentation for this class was generated from the following files:

- include/fita.hpp
- src/Filme/fita.cpp

6.6 Locacao Class Reference

Classe responsável por gerenciar as operações de locação de filmes.

```
#include <locacao.hpp>
```

Public Member Functions

- void cadastrarFilme (Filme *filme)

  Cadastra um novo filme no catálogo.
- void removerFilme (std::string codigo)

  Remove um filme do catálogo pelo código.
- void alugarFilmes (const std::string &cpf, std::vector< Filme * > &filmes)

  Realiza o aluguel de filmes para um cliente.
- void emiteReciboAluguel (Cliente *cliente, std::vector< Filme * > &filmes)

  Emite o recibo de aluguel para um cliente.
- void devolverFilmes (Cliente *cliente, int dias)

  Realiza a devolução de filmes por parte do cliente.
- void emiteReciboDevolucao (Cliente *cliente, int dias)

  Emite o recibo de devolução para um cliente.
- void listarFilmesCodigo ()

  Lista os filmes cadastrados no sistema ordenados por código.
- void listarFilmesTitulo ()

  Lista os filmes cadastrados no sistema ordenados por título.
- void cadastrarCliente (const std::string &nome, const std::string &cpf)

  Cadastra um novo cliente no sistema.
- void removerCliente (const std::string &cpf)

  Remove um cliente do sistema pelo CPF.
- void listarClientesCodigo ()

  Lista os clientes cadastrados no sistema ordenados por código.
- void listarClientesNome ()

  Lista os clientes cadastrados no sistema ordenados por nome.
- Cliente * getCliente (const std::string &cpf)

  Obtém um cliente pelo CPF.
- Filme * getFilme (std::string codigo_filme)

  Obtém um filme pelo código.

Protected Attributes

- std::vector< Filme * > _catalogo_filmes
- std::vector< Cliente * > _clientes_cadastrados

6.6.1 Detailed Description

Classe responsável por gerenciar as operações de locação de filmes.

A classe Locacao realiza o cadastro, remoção, aluguel e devolução de filmes, além de manter registros dos filmes disponíveis no catálogo e dos clientes cadastrados.

6.6.2 Member Function Documentation

6.6.2.1 alugarFilmes()

```
void Locacao::alugarFilmes (
            const std::string & cpf,
            std::vector< Filme * > & filmes_a_alugar )
```

Realiza o aluguel de filmes para um cliente.

Realiza a locação de filmes para um cliente.

Parameters

cpf	CPF do cliente.
filmes	Vetor de ponteiros para os filmes a serem alugados.
cpf	CPF do cliente que deseja alugar os filmes.
filmes_a_alugar	Vetor de ponteiros para os filmes que o cliente deseja alugar.

6.6.2.2 cadastrarCliente()

```
void Locacao::cadastrarCliente (
            const std::string & nome,
            const std::string & cpf )
```

Cadastra um novo cliente no sistema.

Parameters

nome	Nome do cliente.
cpf	CPF do cliente.
nome	Nome do cliente a ser cadastrado.
cpf	CPF do cliente a ser cadastrado.

6.6.2.3 cadastrarFilme()

```
void Locacao::cadastrarFilme (
            Filme * filme )
```

Cadastra um novo filme no catálogo.

Parameters

filme	Ponteiro para o filme a ser cadastrado.

6.6.2.4 devolverFilmes()

```
void Locacao::devolverFilmes (
            Cliente * cliente,
            int dias )
```

Realiza a devolução de filmes por parte do cliente.

Realiza a devolução de filmes por um cliente.

Parameters

cliente	Ponteiro para o cliente.
dias	Número de dias pelos quais os filmes foram alugados.
cliente	Ponteiro para o cliente que está devolvendo os filmes.
dias	Número de dias pelos quais os filmes foram alugados.

6.6.2.5 emiteReciboAluguel()

```
void Locacao::emiteReciboAluguel (
            Cliente * cliente,
            std::vector< Filme * > & filmes )
```

Emite o recibo de aluguel para um cliente.

Emite um recibo de aluguel para o cliente.

Parameters

cliente	Ponteiro para o cliente.
filmes	Vetor de ponteiros para os filmes alugados.
cliente	Ponteiro para o cliente que alugou os filmes.
filmes	Vetor de ponteiros para os filmes alugados.

6.6.2.6 emiteReciboDevolucao()

```
void Locacao::emiteReciboDevolucao (
            Cliente * cliente,
            int dias )
```

Emite o recibo de devolução para um cliente.

Emite um recibo de devolução para o cliente.

Parameters

cliente	Ponteiro para o cliente.
dias	Número de dias pelos quais os filmes foram alugados.
cliente	Ponteiro para o cliente que devolveu os filmes.
dias	Número de dias pelos quais os filmes foram alugados.

6.6.2.7 getCliente()

```
Cliente * Locacao::getCliente (
            const std::string & cpf )
```

Obtém um cliente pelo CPF.

Retorna um ponteiro para um Cliente com base no CPF fornecido.

Parameters

cpf	CPF do cliente.

Returns

Ponteiro para o cliente encontrado, ou nullptr se não encontrado.

Parameters

cpf	CPF do cliente a ser encontrado.

Returns

Ponteiro para o Cliente encontrado ou nullptr se não for encontrado.

6.6.2.8 getFilme()

```
Filme * Locacao::getFilme (
            std::string codigo_filme )
```

Obtém um filme pelo código.

Retorna um ponteiro para um Filme com base no código fornecido.

Parameters

codigo_filme	Código do filme.

Returns

Ponteiro para o filme encontrado, ou nullptr se não encontrado.

Parameters

codigo_filme	Código do filme a ser encontrado.

Returns

Ponteiro para o Filme encontrado ou nullptr se não for encontrado.

6.6.2.9 listarClientesCodigo()

```
void Locacao::listarClientesCodigo ( )
```

Lista os clientes cadastrados no sistema ordenados por código.

Lista os clientes ordenados por código.

6.6.2.10 listarClientesNome()

```
void Locacao::listarClientesNome ( )
```

Lista os clientes cadastrados no sistema ordenados por nome.

Lista os clientes ordenados por nome.

6.6.2.11 listarFilmesCodigo()

```
void Locacao::listarFilmesCodigo ( )
```

Lista os filmes cadastrados no sistema ordenados por código.

Lista os filmes ordenados por código.

6.6.2.12 listarFilmesTitulo()

```
void Locacao::listarFilmesTitulo ( )
```

Lista os filmes cadastrados no sistema ordenados por título.

Lista os filmes ordenados por título.

6.6.2.13 removerCliente()

```
void Locacao::removerCliente (
            const std::string & cpf )
```

Remove um cliente do sistema pelo CPF.

Remove um cliente do sistema com base no CPF fornecido.

Parameters

cpf	CPF do cliente a ser removido.
cpf	CPF do cliente a ser removido.

Exceptions

std::invalid_argument	se o cliente não for encontrado.

6.6.2.14 removerFilme()

```
void Locacao::removerFilme (
            std::string codigo )
```

Remove um filme do catálogo pelo código.

Remove um filme do catálogo com base no código fornecido.

Parameters

codigo	Código do filme a ser removido.
codigo	Código do filme a ser removido.

Exceptions

std::invalid_argument	se o filme não for encontrado.

6.6.3 Member Data Documentation

6.6.3.1 _catalogo_filmes

```
std::vector<Filme*> Locacao::_catalogo_filmes  [protected]
```

Vetor que armazena os filmes disponíveis para locação.

6.6.3.2 _clientes_cadastrados

```
std::vector<Cliente*> Locacao::_clientes_cadastrados  [protected]
```

Vetor que armazena os clientes cadastrados no sistema.

The documentation for this class was generated from the following files:

- include/locacao.hpp
- src/locacao.cpp

6.7 Sistema Class Reference

Classe que representa o sistema de locação de filmes.

```
#include <sistema.hpp>
```

Public Member Functions

- Sistema ()

  Construtor da classe Sistema.
- void displayHelp ()

  Exibe informações de ajuda sobre os comandos disponíveis no sistema.
- void cadastrarFilme (std::deque< std::string > lista_input)

  Realiza a operação de cadastrar um novo filme no sistema.
- void cadastrarCliente (std::deque< std::string > lista_input)

  Realiza a operação de cadastrar um novo cliente no sistema.
- void removerCliente (std::deque< std::string > lista_input)

  Realiza a operação de remover um cliente do sistema.
- void removerFilme (std::deque< std::string > lista_input)

  Realiza a operação de remover um filme do sistema.
- void listarFilmes (std::deque< std::string > lista_input)

  Lista os filmes cadastrados no sistema.
- void listarClientes (std::deque< std::string > lista_input)

  Lista os clientes cadastrados no sistema.
- void devolverFilmes (std::deque< std::string > lista_input)

  Realiza a operação de devolução de filmes por parte do cliente.
- void alugarFilmes (std::deque< std::string > lista_input)

  Realiza a operação de aluguel de filmes para um cliente.
- void lerArquivo (string nome_arquivo)

  Realiza a leitura de informações de arquivo para inicializar o sistema.
- bool gerenciaOpcoes (string comando, std::deque< std::string > lista_input)

  Gerencia o sistema com base no input do usuário.
- std::tuple< std::string, std::deque< std::string > > processaInput (std::string input)

  Recebe o input em linha e retorna uma tupla contendo uma string com a operação a ser realizada e um deque com os comandos restantes.

6.7.1 Detailed Description

Classe que representa o sistema de locação de filmes.

A classe Sistema gerencia as operações do sistema de locação, interagindo com a classe Locacao.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 Sistema()

```
Sistema::Sistema ( )
```

Construtor da classe Sistema.

Inicializa uma instância da classe Sistema.

6.7.3 Member Function Documentation

6.7.3.1 alugarFilmes()

```
void Sistema::alugarFilmes (
            std::deque< std::string > lista_input )
```

Realiza a operação de aluguel de filmes para um cliente.

Aluga filmes para um cliente.

Parameters

lista_input	Lista contendo os inputs da operação selecionada.

Exceptions

std::invalid_argument	se o cliente não for encontrado ou o código do filme for inexistente.

Parameters

lista_input	lista de inputs a serem utilizados pela função

6.7.3.2 cadastrarCliente()

```
void Sistema::cadastrarCliente (
            std::deque< std::string > lista_input )
```

Realiza a operação de cadastrar um novo cliente no sistema.

Solicita informações e cadastra um novo cliente na locadora.

Parameters

lista_input	Lista contendo os inputs da operação selecionada.

Exceptions

std::invalid_argument	se o CPF fornecido for inválido.

Parameters

lista_input	lista de inputs a serem utilizados pela função

6.7.3.3 cadastrarFilme()

```
void Sistema::cadastrarFilme (
            std::deque< std::string > lista_input )
```

Realiza a operação de cadastrar um novo filme no sistema.

Cadastra um novo filme na locadora.

Parameters

lista_input	Lista contendo os inputs da operação selecionada.

Exceptions

std::invalid_argument	se o tipo de filme for inválido.

Parameters

lista_input	lista de inputs a serem utilizados pela função

6.7.3.4 devolverFilmes()

```
void Sistema::devolverFilmes (
            std::deque< std::string > lista_input )
```

Realiza a operação de devolução de filmes por parte do cliente.

Permite que um cliente devolva filmes alugados.

Parameters

lista_input	Lista contendo os inputs da operação selecionada.

Exceptions

std::out_of_range	se nenhum filme estiver alugado para o cliente.
std::invalid_argument	se o cliente não for encontrado.

Parameters

lista_input	lista de inputs a serem utilizados pela função

6.7.3.5 displayHelp()

```
void Sistema::displayHelp ( )
```

Exibe informações de ajuda sobre os comandos disponíveis no sistema.

Exibe o menu de ajuda com os comandos disponíveis.

6.7.3.6 gerenciaOpcoes()

```
bool Sistema::gerenciaOpcoes (
            string comando,
            std::deque< std::string > lista_input )
```

Gerencia o sistema com base no input do usuário.

Direciona para a função adequada dado o comando fornecido.

Parameters

lista_input	Lista contendo os inputs da operação selecionada.

Exceptions

std::invalid_argument	se o comando não existir.

Parameters

lista_input	lista de inputs a serem utilizados pela função

6.7.3.7 lerArquivo()

```
void Sistema::lerArquivo (
            string nome_arquivo )
```

Realiza a leitura de informações de arquivo para inicializar o sistema.

Lê informações de um arquivo e realiza operações no sistema da locadora.

Parameters

nome_arquivo	Nome do arquivo a ser lido.

Exceptions

std::invalid_argument	se o arquivo não existir.

Parameters

nome_arquivo	string com o nome do arquivo a ser aberto para leitura no sistema

6.7.3.8 listarClientes()

```
void Sistema::listarClientes (
            std::deque< std::string > lista_input )
```

Lista os clientes cadastrados no sistema.

Lista os clientes cadastrados na locadora por CPF ou nome.

Parameters

lista_input	Lista contendo os inputs da operação selecionada.

Exceptions

std::invalid_argument	se a opção de listagem for inválida.

Parameters

lista_input	lista de inputs a serem utilizados pela função

6.7.3.9 listarFilmes()

```
void Sistema::listarFilmes (
            std::deque< std::string > lista_input )
```

Lista os filmes cadastrados no sistema.

Lista os filmes disponíveis na locadora por código ou título.

Parameters

lista_input	Lista contendo os inputs da operação selecionada.

Exceptions

std::invalid_argument	se a opção de listagem for inválida.

Parameters

lista_input	lista de inputs a serem utilizados pela função

6.7.3.10 processaInput()

```
std::tuple< std::string, std::deque< std::string > > Sistema::processaInput (
            std::string input )
```

Recebe o input em linha e retorna uma tupla contendo uma string com a operação a ser realizada e um deque com os comandos restantes.

Lê uma lista de inputs em formato de string e os retorna, diferenciando o primeiro que é um comando.

Parameters

input	String contendo o input do usuario.
input	string com o input inserido pelo usuário.

6.7.3.11 removerCliente()

```
void Sistema::removerCliente (
            std::deque< std::string > lista_input )
```

Realiza a operação de remover um cliente do sistema.

Remove um cliente da locadora com base no CPF.

Parameters

lista_input	Lista contendo os inputs da operação selecionada.

Exceptions

std::out_of_range	se nenhum filme estiver alugado para o cliente.

Parameters

lista_input	lista de inputs a serem utilizados pela função

6.7.3.12 removerFilme()

```
void Sistema::removerFilme (
            std::deque< std::string > lista_input )
```

Realiza a operação de remover um filme do sistema.

Remove um filme da locadora com base no código.

Parameters

lista_input	Lista contendo os inputs da operação selecionada.

Exceptions

std::invalid_argument	se nenhum filme com os códigos fornecidos for encontrado.

Parameters

lista_input	lista de inputs a serem utilizados pela função

The documentation for this class was generated from the following files:

- include/sistema.hpp
- src/sistema.cpp

Chapter 7

File Documentation

7.1 CRC.md File Reference

7.2 include/bluray.hpp File Reference

Declaração da classe Bluray, seus métodos e atributos.

```
#include <iostream>
#include <string>
#include "filme.hpp"
```

Classes

- class Bluray

7.2.1 Detailed Description

Declaração da classe Bluray, seus métodos e atributos.

Este arquivo define a classe Bluray, que representa um filme no formato Blu-ray. A classe herda da classe Filme e adiciona funcionalidades específicas ao Blu-ray, como a gestão de aparelhos Blu-ray e cálculo de valor de locação.

7.3 bluray.hpp

Go to the documentation of this file.

```
00001 #ifndef BLURAY_HPP
00002 #define BLURAY_HPP
00003
00004 #include <iostream>
00005 #include <string>
00006
00007 #include "filme.hpp"
00008
00022 class Bluray : public Filme {
00023 private:
00024     static int quantidadeAparelhos;
```

```
00025     bool playerAlugado;
00027 public:
00034     Bluray(const std::string& titulo, std::string codigo, int unidadesDisponiveis);
00035
00040     static int getQuantidadeAparelhosDisponiveis();
00041
00046     static void setQuantidadeAparelhos(int quantidade);
00047
00054     void alugarAparelho();
00055
00060     void devolverAparelho();
00061
00068     double calcularValorLocacao(int dias) const override;
00069 };
00070
00071 #endif // BLURAY_HPP
```

7.4 include/cliente.hpp File Reference

Declaração da classe Cliente, seus métodos e atributos.

```
#include <iostream>
#include <vector>
#include "filme.hpp"
```

Classes

- class Cliente

7.4.1 Detailed Description

Declaração da classe Cliente, seus métodos e atributos.

Este arquivo define a classe Cliente, que representa um cliente do sistema de locação de filmes.

7.5 cliente.hpp

Go to the documentation of this file.

```
00001 #ifndef CLIENTE_H
00002 #define CLIENTE_H
00003
00004 #include <iostream>
00005 #include <vector>
00006
00007 #include "filme.hpp"
00008
00009 using namespace std;
00010
00026 class Cliente
00027 {
00028 private:
00029     string _nome;
00030     string _cpf;
00031     int _pontos_fidelidade;
00032     bool _alugou_aparelho;
00033     vector <Filme*> _filmes_alugados;
00035 public:
00041     Cliente(string nome, string cpf);
00042
00046     ~Cliente();
00047
00051     void mostrarInfo();
```

```
00052
00057       void adicionarPontos(int pontos);
00058
00062       void usarPontos();
00063
00064       // Getters
00065
00070       string getNome();
00071
00076       bool getAparelhoAlugado();
00077
00082       string getCpf();
00083
00088       vector<Filme*>& getFilmesAlugados();
00089
00094       int getPontos();
00095
00100       void adicionarFilmeAlugado(Filme* filme);
00101
00105       void devolverFilmesAlugados();
00106
00110       void alugarAparelhoBluray();
00111
00115       void devolverAparelhoBluray();
00116 };
00117
00118 #endif // CLIENTE_H
```

7.6 include/dvd.hpp File Reference

Declaração da classe DVD, seus métodos e atributos.

```
#include "filme.hpp"
```

Classes

- class DVD

Enumerations

- enum class Categoria { Lancamento , Estoque , Promocao }

7.6.1 Detailed Description

Declaração da classe DVD, seus métodos e atributos.

Este arquivo define a classe DVD, que representa um filme no formato DVD. A classe herda da classe Filme e adiciona funcionalidades específicas ao DVD, como a definição da categoria e cálculo de valor de locação.

7.6.2 Enumeration Type Documentation

7.6.2.1 Categoria

```
enum class Categoria  [strong]
```

Enumerator

Lancamento	Categoria de lançamento do DVD.
Estoque	Categoria de estoque do DVD.
Promocao	Categoria de promoção do DVD.

7.7 dvd.hpp

Go to the documentation of this file.

```
00001 #ifndef DVD_HPP
00002 #define DVD_HPP
00003
00004 #include "filme.hpp"
00005
00018 enum class Categoria {
00019     Lancamento,
00020     Estoque,
00021     Promocao
00022 };
00023
00024 class DVD : public Filme {
00025 protected:
00026     Categoria categoria;
00028 public:
00036     DVD(const std::string& titulo, std::string codigo, int unidadesDisponiveis, Categoria _categoria);
00037
00044     double calcularValorLocacao(int dias) const override;
00045 };
00046
00047 #endif // DVD_HPP
```

7.8 include/filme.hpp File Reference

Declaração da classe Filme, seus métodos e atributos.

```
#include <iostream>
#include <string>
```

Classes

- class Filme

Functions

- std::ostream & operator<< (std::ostream &os, const Filme::Tipo &tipo)

  Sobrecarga do operador de saída para imprimir o tipo do filme.

7.8.1 Detailed Description

Declaração da classe Filme, seus métodos e atributos.

Este arquivo define a classe base Filme, que representa um filme genérico. As classes específicas (DVD, BluRay, Fita) herdam desta classe base.

7.8.2 Function Documentation

7.8.2.1 operator<<()

```
std::ostream & operator<< (
            std::ostream & os,
            const Filme::Tipo & tipo )  [inline]
```

Sobrecarga do operador de saída para imprimir o tipo do filme.

Parameters

os	Objeto de fluxo de saída.
tipo	Tipo do filme (DVD, BluRay, Fita).

Returns

O objeto de fluxo de saída.

7.9 filme.hpp

Go to the documentation of this file.

```
00001 #ifndef FILME_HPP
00002 #define FILME_HPP
00003
00004 #include <iostream>
00005 #include <string>
00006
00021 class Filme {
00022 public:
00026     enum class Tipo {
00027         DVD,
00028         BluRay,
00029         Fita
00030     };
00031
00032 private:
00033     std::string titulo;
00034     std::string codigo;
00035     int unidadesDisponiveis;
00036     Tipo tipo;
00038 public:
00046     Filme(const std::string& titulo, std::string codigo, Tipo tipo, int unidadesDisponiveis);
00047
00051     virtual ~Filme();
00052
00057     std::string getCodigo() const;
00058
00063     const std::string& getTitulo() const;
00064
00069     int getUnidadesDisponiveis() const;
00070
00075     Tipo getTipo() const;
00076
00081     void adicionarUnidadesDisponiveis(int quantidade);
00082
00086     void removerUnidadeDisponivel();
00087
00094     virtual double calcularValorLocacao(int dias) const = 0;
00095
00102     friend std::ostream& operator«(std::ostream& os, const Filme::Tipo& tipo);
00103 };
00104
00111 inline std::ostream& operator«(std::ostream& os, const Filme::Tipo& tipo) {
00112     switch (tipo) {
00113         case Filme::Tipo::DVD:
00114             os « "DVD";
00115             break;
```

```
00116         case Filme::Tipo::BluRay:
00117             os « "BluRay";
00118             break;
00119         case Filme::Tipo::Fita:
00120             os « "Fita";
00121             break;
00122     }
00123     return os;
00124 }
00125
00126 #endif // FILME_HPP
```

7.10 include/fita.hpp File Reference

Declaração da classe Fita, seus métodos e atributos.

```
#include "filme.hpp"
```

Classes

- class Fita

7.10.1 Detailed Description

Declaração da classe Fita, seus métodos e atributos.

Este arquivo define a classe Fita, que representa um filme no formato de fita. A classe herda da classe Filme e adiciona funcionalidades específicas às fitas, como rebobinamento e cálculo de valor de locação.

7.11 fita.hpp

Go to the documentation of this file.

```
00001 #ifndef FITA_HPP
00002 #define FITA_HPP
00003
00004 #include "filme.hpp"
00005
00018 class Fita : public Filme {
00019 private:
00020     bool rebobinada;
00022 public:
00030     Fita(const std::string& titulo, std::string codigo, int quantidade, bool _rebobinada);
00031
00036     bool estaRebobinada() const;
00037
00041     void rebobinar();
00042
00048     double calcularValorLocacao(int dias) const override;
00049 };
00050
00051 #endif // FITA_HPP
```

7.12 include/formatacao.hpp File Reference

Funções para formatação de strings.

```
#include <iostream>
#include <string>
#include <algorithm>
#include <cctype>
#include <iomanip>
#include <deque>
```

Functions

- std::string toLowerCase (const std::string &input)

  Converte uma string para letras minúsculas.
- std::string toUpperCase (const std::string &input)

  Converte uma string para letras maiúsculas.
- std::string firstUpper (const std::string &input)

  Converte a primeira letra de uma string para maiúscula.
- std::string retornaStringFormatada (std::deque< std::string > palavra)

  Torna um array de strings uma string só. Converte a primeira letra de uma string para maiúscula.

7.12.1 Detailed Description

Funções para formatação de strings.

7.12.2 Function Documentation

7.12.2.1 firstUpper()

```
std::string firstUpper (
            const std::string & input )
```

Converte a primeira letra de uma string para maiúscula.

Parameters

input	A string a ser convertida.

Returns

A string com a primeira letra em maiúscula.

Converte a primeira letra de uma string para maiúscula.

Parameters

input	A string a ser convertida.

Returns

A string com a primeira letra de cada palavra em maiúscula.

7.12.2.2 retornaStringFormatada()

```
std::string retornaStringFormatada (
            std::deque< std::string > palavra )
```

Torna um array de strings uma string só. Converte a primeira letra de uma string para maiúscula.

Parameters

input	O deque de strings a ser unificado.

Returns

A string formatada com a primeira letra em maiúscula.

7.12.2.3 toLowerCase()

```
std::string toLowerCase (
            const std::string & input )
```

Converte uma string para letras minúsculas.

Parameters

input	A string a ser convertida.

Returns

A string convertida para letras minúsculas.

Converte uma string para letras minúsculas.

Parameters

input	A string a ser convertida.

Returns

A string convertida para minúsculas.

7.12.2.4 toUpperCase()

```
std::string toUpperCase (
            const std::string & input )
```

Converte uma string para letras maiúsculas.

Parameters

input	A string a ser convertida.

Returns

A string convertida para letras maiúsculas.

Converte uma string para letras maiúsculas.

Parameters

input	A string a ser convertida.

Returns

A string convertida para maiúsculas.

7.13 formatacao.hpp

Go to the documentation of this file.

```
00001 #ifndef FORMATACAO_H
00002 #define FORMATACAO_H
00003
00004 #include <iostream>
00005 #include <string>
00006 #include <algorithm>
00007 #include <cctype>
00008 #include <iomanip>
00009 #include <deque>
00010
00022 std::string toLowerCase(const std::string& input);
00023
00030 std::string toUpperCase(const std::string& input);
00031
00038 std::string firstUpper(const std::string& input);
00039
00046 std::string retornaStringFormatada(std::deque<std::string> palavra);
00047
00048
00049 #endif // FORMATACAO_H
```

7.14 include/locacao.hpp File Reference

Declaração da classe Locacao, seus métodos e atributos.

```
#include <iostream>
#include <vector>
#include "filme.hpp"
#include "cliente.hpp"
```

Classes

- class Locacao

  Classe responsável por gerenciar as operações de locação de filmes.

7.14.1 Detailed Description

Declaração da classe Locacao, seus métodos e atributos.

7.15 locacao.hpp

Go to the documentation of this file.

```
00001 #ifndef LOCACAO_HPP
00002 #define LOCACAO_HPP
00003
00004 #include <iostream>
00005 #include <vector>
00006
00007 #include "filme.hpp"
00008 #include "cliente.hpp"
00009
00022 class Locacao {
00023 protected:
00024     std::vector<Filme*> _catalogo_filmes;
00025     std::vector<Cliente*> _clientes_cadastrados;
00027 public:
00033     void cadastrarFilme(Filme* filme);
00034
00040     void removerFilme(std::string codigo);
00041
00048     void alugarFilmes(const std::string& cpf, std::vector<Filme*>& filmes);
00049
00056     void emiteReciboAluguel(Cliente* cliente, std::vector<Filme*>& filmes);
00057
00064     void devolverFilmes(Cliente* cliente, int dias);
00065
00072     void emiteReciboDevolucao(Cliente* cliente, int dias);
00073
00077     void listarFilmesCodigo();
00078
00082     void listarFilmesTitulo();
00083
00090     void cadastrarCliente(const std::string& nome, const std::string& cpf);
00091
00097     void removerCliente(const std::string& cpf);
00098
00102     void listarClientesCodigo();
00103
00107     void listarClientesNome();
00108
00115     Cliente* getCliente(const std::string& cpf);
00116
00123     Filme* getFilme(std::string codigo_filme);
00124 };
00125
00126 #endif // LOCACAO_HPP
```

7.16 include/sistema.hpp File Reference

Declaração da classe Sistema e seus métodos.

```
#include "locacao.hpp"
#include "filme.hpp"
```

Classes

- class Sistema

  Classe que representa o sistema de locação de filmes.

7.16.1 Detailed Description

Declaração da classe Sistema e seus métodos.

7.17 sistema.hpp

Go to the documentation of this file.

```
00001 #ifndef SISTEMA_H
00002 #define SISTEMA_H
00003
00004 #include "locacao.hpp"
00005 #include "filme.hpp"
00006
00017 class Sistema {
00018 private:
00019     Locacao _locacao;
00020     vector<Filme*> filmes;
00022 public:
00028     Sistema();
00029
00033     void displayHelp();
00034
00039     void cadastrarFilme(std::deque<std::string> lista_input);
00040
00045     void cadastrarCliente(std::deque<std::string> lista_input);
00046
00051     void removerCliente(std::deque<std::string> lista_input);
00052
00057     void removerFilme(std::deque<std::string> lista_input);
00058
00063     void listarFilmes(std::deque<std::string> lista_input);
00064
00069     void listarClientes(std::deque<std::string> lista_input);
00070
00075     void devolverFilmes(std::deque<std::string> lista_input);
00076
00081     void alugarFilmes(std::deque<std::string> lista_input);
00082
00087     void lerArquivo(string nome_arquivo);
00088
00093     bool gerenciaOpcoes(string comando, std::deque<std::string> lista_input);
00094
00100     std::tuple<std::string, std::deque<std::string>> processaInput(std::string input);
00101 };
00102
00103 #endif // SISTEMA_H
```

7.18 README.md File Reference

7.19 src/Cliente/cliente.cpp File Reference

Implementação dos métodos da classe Cliente.

```
#include "cliente.hpp"
#include <algorithm>
```

7.19.1 Detailed Description

Implementação dos métodos da classe Cliente.

7.20 src/Filme/bluray.cpp File Reference

Implementação dos métodos da classe Bluray, uma classe filha da classe Filme.

```
#include "bluray.hpp"
#include <stdexcept>
```

7.20.1 Detailed Description

Implementação dos métodos da classe Bluray, uma classe filha da classe Filme.

7.21 src/Filme/dvd.cpp File Reference

Implementação dos métodos da classe DVD, uma classe filha da classe Filme.

```
#include "dvd.hpp"
```

7.21.1 Detailed Description

Implementação dos métodos da classe DVD, uma classe filha da classe Filme.

7.22 src/Filme/filme.cpp File Reference

Implementação dos métodos da classe Filme, uma classe mãe para os diferentes tipos de filmes.

```
#include "filme.hpp"
```

7.22.1 Detailed Description

Implementação dos métodos da classe Filme, uma classe mãe para os diferentes tipos de filmes.

7.23 src/Filme/fita.cpp File Reference

Implementação dos métodos da classe Fita, uma classe filha da classe Filme, representando uma fita de vídeo.

```
#include "fita.hpp"
```

7.23.1 Detailed Description

Implementação dos métodos da classe Fita, uma classe filha da classe Filme, representando uma fita de vídeo.

7.24 src/formatacao.cpp File Reference

Implementação das funções de formatação de strings.

```
#include "formatacao.hpp"
#include <algorithm>
#include <cctype>
```

Functions

- std::string toLowerCase (const std::string &input)
  Converte uma string para minúsculas.
- std::string toUpperCase (const std::string &input)
  Converte uma string para maiúsculas.
- std::string firstUpper (const std::string &input)
  Converte a primeira letra de cada palavra para maiúscula.
- std::string retornaStringFormatada (std::deque< std::string > palavra)
  Torna um array de strings uma string só. Converte a primeira letra de uma string para maiúscula.

7.24.1 Detailed Description

Implementação das funções de formatação de strings.

7.24.2 Function Documentation

7.24.2.1 firstUpper()

```
std::string firstUpper (
            const std::string & input )
```

Converte a primeira letra de cada palavra para maiúscula.

Converte a primeira letra de uma string para maiúscula.

Parameters

input	A string a ser convertida.

Returns

A string com a primeira letra de cada palavra em maiúscula.

7.24.2.2 retornaStringFormatada()

```
std::string retornaStringFormatada (
            std::deque< std::string > palavra )
```

Torna um array de strings uma string só. Converte a primeira letra de uma string para maiúscula.

Parameters

input	O deque de strings a ser unificado.

Returns

A string formatada com a primeira letra em maiúscula.

7.24.2.3 toLowerCase()

```
std::string toLowerCase (
            const std::string & input )
```

Converte uma string para minúsculas.

Converte uma string para letras minúsculas.

Parameters

input	A string a ser convertida.

Returns

A string convertida para minúsculas.

7.24.2.4 toUpperCase()

```
std::string toUpperCase (
            const std::string & input )
```

Converte uma string para maiúsculas.

Converte uma string para letras maiúsculas.

Parameters

input	A string a ser convertida.

Returns

A string convertida para maiúsculas.

7.25 src/locacao.cpp File Reference

Implementação dos métodos da classe Locacao.

```
#include "locacao.hpp"
#include "dvd.hpp"
#include "bluray.hpp"
#include "fita.hpp"
#include <algorithm>
#include <iostream>
```

7.25.1 Detailed Description

Implementação dos métodos da classe Locacao.

7.26 src/main.cpp File Reference

Arquivo principal para o programa Locadora PDS.

```
#include <iostream>
#include <string>
#include <deque>
#include <iomanip>
#include "formatacao.hpp"
#include "sistema.hpp"
```

Functions

- int main ()

  Função principal para o programa Locadora PDS.

7.26.1 Detailed Description

Arquivo principal para o programa Locadora PDS.

7.26.2 Function Documentation

7.26.2.1 main()

```
int main ( )
```

Função principal para o programa Locadora PDS.

Returns

0 em execução bem-sucedida.

7.27 src/sistema.cpp File Reference

Implementação dos métodos da classe Sistema.

```
#include <iostream>
#include <tuple>
#include <string>
#include <fstream>
#include <deque>
#include <algorithm>
#include <cctype>
#include <iomanip>
#include <sstream>
#include <vector>
#include "locacao.hpp"
#include "dvd.hpp"
#include "bluray.hpp"
#include "fita.hpp"
#include "formatacao.hpp"
#include "sistema.hpp"
```

7.27.1 Detailed Description

Implementação dos métodos da classe Sistema.

7.28 test/Testcliente.cpp File Reference

```
#include "filme.hpp"
#include "doctest.h"
#include "dvd.hpp"
#include "cliente.hpp"
```

Macros

- #define DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN

Functions

- TEST_CASE ("Testando Getters")
- TEST_CASE ("Testando sistema de pontos")
- TEST_CASE ("Testando vetor de filmes")

7.28.1 Macro Definition Documentation

7.28.1.1 DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN

```
#define DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN
```

7.28.2 Function Documentation

7.28.2.1 TEST_CASE() [1/3]

```
TEST_CASE (
            "Testando Getters"  )
```

7.28.2.2 TEST_CASE() [2/3]

```
TEST_CASE (
            "Testando sistema de pontos"  )
```

7.28.2.3 TEST_CASE() [3/3]

```
TEST_CASE (
            "Testando vetor de filmes"  )
```

7.29 test/Testelocacao.cpp File Reference

```
#include "doctest.h"
#include "locacao.hpp"
#include "dvd.hpp"
```

Macros

- #define DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN

7.29.1 Macro Definition Documentation

7.29.1.1 DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN

```
#define DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN
```

7.30 test/Testfilme.cpp File Reference

```
#include "doctest.h"
#include "dvd.hpp"
#include "fita.hpp"
#include "bluray.hpp"
```

Macros

- #define DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN

Functions

- TEST_CASE ("Testando Getters")
- TEST_CASE ("Testando unidades disponiveis")
- TEST_CASE ("Testando classe dvd")
- TEST_CASE ("Testando classe fita")
- TEST_CASE ("Testando classe bluray")

7.30.1 Macro Definition Documentation

7.30.1.1 DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN

```
#define DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN
```

7.30.2 Function Documentation

7.30.2.1 TEST_CASE() [1/5]

```
TEST_CASE (
            "Testando classe bluray"  )
```

7.30.2.2 TEST_CASE() [2/5]

```
TEST_CASE (
            "Testando classe dvd"  )
```

7.30.2.3 TEST_CASE() [3/5]

```
TEST_CASE (
            "Testando classe fita"  )
```

7.30.2.4 TEST_CASE() [4/5]

```
TEST_CASE (
            "Testando Getters"  )
```

7.30.2.5 TEST_CASE() [5/5]

```
TEST_CASE (
            "Testando unidades disponiveis"  )
```