

TP - Sistema da Locadora

1.0

Generated by Doxygen 1.9.8

1 Sistema da Locadora PDS	1
1.1 Descrição do Projeto	1
1.2 Equipe	1
1.3 Contexto do Projeto	1
1.4 Tecnologias Utilizadas	1
1.5 Estrutura do Projeto	2
1.6 Funcionalidades Principais	2
1.7 Funcionalidades Extras	2
1.8 Diferenças de input	2
1.9 Aprendizados	3
1.10 Documentação	3
1.11 Github da Equipe	3
2 pds2-projeto-final	5
3 Hierarchical Index	7
3.1 Class Hierarchy	7
4 Class Index	9
4.1 Class List	9
5 File Index	11
5.1 File List	11
6 Class Documentation	13
6.1 Bluray Class Reference	13
6.1.1 Detailed Description	14
6.1.2 Constructor & Destructor Documentation	14
6.1.2.1 Bluray()	14
6.1.3 Member Function Documentation	15
6.1.3.1 alugarAparelho()	15
6.1.3.2 calcularValorLocacao()	15
6.1.3.3 devolverAparelho()	16
6.1.3.4 getQuantidadeAparelhosDisponiveis()	16
6.1.3.5 setQuantidadeAparelhos()	16
6.2 Cliente Class Reference	17
6.2.1 Detailed Description	17
6.2.2 Constructor & Destructor Documentation	18
6.2.2.1 Cliente()	18
6.2.2.2 ~Cliente()	18
6.2.3 Member Function Documentation	18
6.2.3.1 adicionarFilmeAlugado()	18
6.2.3.2 adicionarPontos()	18
6.2.3.3 alugarAparelhoBluray()	19

6.2.3.4 devolverAparelhoBluray()	19
6.2.3.5 devolverFilmesAlugados()	19
6.2.3.6 getAparelhoAlugado()	19
6.2.3.7 getCpf()	19
6.2.3.8 getFilmesAlugados()	20
6.2.3.9 getNome()	20
6.2.3.10 getPontos()	20
6.2.3.11 mostrarInfo()	20
6.2.3.12 usarPontos()	21
6.3 DVD Class Reference	21
6.3.1 Detailed Description	22
6.3.2 Constructor & Destructor Documentation	22
6.3.2.1 DVD()	22
6.3.3 Member Function Documentation	23
6.3.3.1 calcularValorLocacao()	23
6.3.4 Member Data Documentation	23
6.3.4.1 categoria	23
6.4 Filme Class Reference	24
6.4.1 Detailed Description	24
6.4.2 Member Enumeration Documentation	25
6.4.2.1 Tipo	25
6.4.3 Constructor & Destructor Documentation	25
6.4.3.1 Filme()	25
6.4.3.2 ~Filme()	26
6.4.4 Member Function Documentation	26
6.4.4.1 adicionarUnidadesDisponiveis()	26
6.4.4.2 calcularValorLocacao()	26
6.4.4.3 getCodigo()	27
6.4.4.4 getTipo()	27
6.4.4.5 getTitulo()	27
6.4.4.6 getUnidadesDisponiveis()	28
6.4.4.7 removerUnidadeDisponivel()	28
6.4.5 Friends And Related Symbol Documentation	28
6.4.5.1 operator<<	28
6.5 Fita Class Reference	29
6.5.1 Detailed Description	29
6.5.2 Constructor & Destructor Documentation	30
6.5.2.1 Fita()	30
6.5.3 Member Function Documentation	30
6.5.3.1 calcularValorLocacao()	30
6.5.3.2 estaRebobinada()	31
6.5.3.3 rebobinar()	31

6.6 Locacao Class Reference	31
6.6.1 Detailed Description	32
6.6.2 Member Function Documentation	33
6.6.2.1 alugarFilmes()	33
6.6.2.2 cadastrarCliente()	33
6.6.2.3 cadastrarFilme()	33
6.6.2.4 devolverFilmes()	34
6.6.2.5 emiteReciboAluguel()	34
6.6.2.6 emiteReciboDevolucao()	34
6.6.2.7 getCliente()	35
6.6.2.8 getFilme()	35
6.6.2.9 getvectorClientes()	36
6.6.2.10 getvectorFilmes()	36
6.6.2.11 listarClientesCodigo()	36
6.6.2.12 listarClientesNome()	36
6.6.2.13 listarFilmesCodigo()	36
6.6.2.14 listarFilmesTitulo()	37
6.6.2.15 removerCliente()	37
6.6.2.16 removerFilme()	37
6.6.3 Member Data Documentation	38
6.6.3.1 _catalogo_filmes	38
6.6.3.2 _clientes_cadastrados	38
6.7 Sistema Class Reference	38
6.7.1 Detailed Description	39
6.7.2 Member Function Documentation	39
6.7.2.1 alugarFilmes()	39
6.7.2.2 cadastrarCliente()	39
6.7.2.3 cadastrarFilme()	40
6.7.2.4 devolverFilmes()	40
6.7.2.5 displayHelp()	41
6.7.2.6 gerenciaOpcoes()	41
6.7.2.7 lerArquivo()	41
6.7.2.8 listarClientes()	42
6.7.2.9 listarFilmes()	42
6.7.2.10 processaInput()	43
6.7.2.11 removerCliente()	43
6.7.2.12 removerFilme()	43
7 File Documentation	45
7.1 CRC.md File Reference	45
7.2 include/bluray.hpp File Reference	45
7.2.1 Detailed Description	45

7.3 bluray.hpp	45
7.4 include/cliente.hpp File Reference	46
7.4.1 Detailed Description	46
7.5 cliente.hpp	46
7.6 include/dvd.hpp File Reference	47
7.6.1 Detailed Description	47
7.6.2 Enumeration Type Documentation	47
7.6.2.1 Categoria	47
7.7 dvd.hpp	48
7.8 include/filme.hpp File Reference	48
7.8.1 Detailed Description	48
7.8.2 Function Documentation	49
7.8.2.1 operator<<()	49
7.9 filme.hpp	49
7.10 include/fita.hpp File Reference	50
7.10.1 Detailed Description	50
7.11 fita.hpp	50
7.12 include/formatacao.hpp File Reference	51
7.12.1 Detailed Description	51
7.12.2 Function Documentation	51
7.12.2.1 firstUpper()	51
7.12.2.2 retornaStringFormatada()	52
7.12.2.3 toLowerCase()	52
7.12.2.4 toUpperCase()	53
7.13 formatacao.hpp	53
7.14 include/locacao.hpp File Reference	54
7.14.1 Detailed Description	54
7.15 locacao.hpp	54
7.16 include/sistema.hpp File Reference	55
7.16.1 Detailed Description	55
7.17 sistema.hpp	55
7.18 README.md File Reference	56
7.19 src/Cliente/cliente.cpp File Reference	56
7.19.1 Detailed Description	56
7.20 src/Filme/bluray.cpp File Reference	56
7.20.1 Detailed Description	56
7.21 src/Filme/dvd.cpp File Reference	56
7.21.1 Detailed Description	56
7.22 src/Filme/filme.cpp File Reference	56
7.22.1 Detailed Description	57
7.23 src/Filme/fita.cpp File Reference	57
7.23.1 Detailed Description	57

7.24 src/formatacao.cpp File Reference	57
7.24.1 Detailed Description	57
7.24.2 Function Documentation	57
7.24.2.1 firstUpper()	57
7.24.2.2 retornaStringFormatada()	58
7.24.2.3 toLowerCase()	58
7.24.2.4 toUpperCase()	58
7.25 src/locacao.cpp File Reference	59
7.25.1 Detailed Description	59
7.26 src/main.cpp File Reference	59
7.26.1 Detailed Description	59
7.26.2 Function Documentation	60
7.26.2.1 main()	60
7.27 src/sistema.cpp File Reference	60
7.27.1 Detailed Description	60
7.28 test/Testcliente.cpp File Reference	60
7.28.1 Macro Definition Documentation	61
7.28.1.1 DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN	61
7.28.2 Function Documentation	61
7.28.2.1 TEST_CASE() [1/3]	61
7.28.2.2 TEST_CASE() [2/3]	61
7.28.2.3 TEST_CASE() [3/3]	61
7.29 test/Testlocacao.cpp File Reference	61
7.29.1 Macro Definition Documentation	62
7.29.1.1 DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN	62
7.29.2 Function Documentation	62
7.29.2.1 TEST_CASE() [1/4]	62
7.29.2.2 TEST_CASE() [2/4]	62
7.29.2.3 TEST_CASE() [3/4]	62
7.29.2.4 TEST_CASE() [4/4]	62
7.30 test/Testfilme.cpp File Reference	62
7.30.1 Macro Definition Documentation	63
7.30.1.1 DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN	63
7.30.2 Function Documentation	63
7.30.2.1 TEST_CASE() [1/5]	63
7.30.2.2 TEST_CASE() [2/5]	63
7.30.2.3 TEST_CASE() [3/5]	63
7.30.2.4 TEST_CASE() [4/5]	63
7.30.2.5 TEST_CASE() [5/5]	63

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 Funcionalidades Extras

As seguintes funcionalidades extras foram implementadas:

- Leitura de arquivo pode ser utilizada para qualquer operação, não apenas cadastrar filme.
- Implementação de pontuação de fidelidade, onde cada aluguel de filme vale 1 ponto e a cada 5 pontos adquiridos o cliente ganha um desconto de R\$10,00 na hora da devolução.
- Implementação do aluguel de blurays.
- Implementação do aluguel de aparelho leitor de bluray.

1.8 Diferenças de input

Para conseguirmos trabalhar com nomes compostos para filmes, mudamos a ordem dos parâmetros nos inputs de Cadastrar [Filme](#). A ordem é:

- CF <tipo D|B|F> (dvd, bluray ou fita) <quantidade> <codigo> <categoria> (para dvd e bluray) <nome_do_filme> E quando se aluga um filme do tipo bluray, pode se adicionar o parâmetro ALUGAR_APARELHO entre os parâmetros de categoria e nome do filme para solicitar o aluguel de um aparelho (que custa R\$20,00). O comando fica assim:
- CF B <quantidade> <codigo> <categoria> <ALUGAR_APARELHO> <nome_do_filme>

1.9 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.

1.10 Documentação

A documentação detalhada do projeto, feita com a ferramenta Doxygen, pode ser encontrada na forma de página web no arquivo "index.html" e no formato PDF no arquivo "refman.pdf".

1.11 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:

Cliente	17
Filme	24
Bluray	13
DVD	21
Fita	29
Locacao	31
Sistema	38

Chapter 4

Class Index

4.1 Class List

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

Bluray	13
Cliente	17
DVD	21
Filme	24
Fita	29
Locacao	
	Classe responsável por gerenciar as operações de locação de filmes	31
Sistema	
	Classe que representa o sistema de locação de filmes	38

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

include/ bluray.hpp	
Declaração da classe Bluray , seus métodos e atributos	45
include/ cliente.hpp	
Declaração da classe Cliente , seus métodos e atributos	46
include/ dvd.hpp	
Declaração da classe DVD , seus métodos e atributos	47
include/ filme.hpp	
Declaração da classe Filme , seus métodos e atributos	48
include/ fita.hpp	
Declaração da classe Fita , seus métodos e atributos	50
include/ formatacao.hpp	
Funções para formatação de strings	51
include/ locacao.hpp	
Declaração da classe Locacao , seus métodos e atributos	54
include/ sistema.hpp	
Declaração da classe Sistema e seus métodos	55
src/ formatacao.cpp	
Implementação das funções de formatação de strings	57
src/ locacao.cpp	
Implementação dos métodos da classe Locacao	59
src/ main.cpp	
Arquivo principal para o programa Locadora PDS	59
src/ sistema.cpp	
Implementação dos métodos da classe Sistema	60
src/Cliente/ cliente.cpp	
Implementação dos métodos da classe Cliente	56
src/Filme/ bluray.cpp	
Implementação dos métodos da classe Bluray , uma classe filha da classe Filme	56
src/Filme/ dvd.cpp	
Implementação dos métodos da classe DVD , uma classe filha da classe Filme	56
src/Filme/ filme.cpp	
Implementação dos métodos da classe Filme , uma classe mãe para os diferentes tipos de filmes	56
src/Filme/ fita.cpp	
Implementação dos métodos da classe Fita , uma classe filha da classe Filme , representando uma fita de vídeo	57
test/ Testcliente.cpp	60
test/ Testelocacao.cpp	61
test/ Testfilme.cpp	62

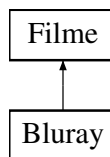
Chapter 6

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

<code>quantidadeAparelhos</code>	Quantidade de aparelhos Blu-ray disponíveis para aluguel.
<code>playerAlugado</code>	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

<code>titulo</code>	Título do filme.
<code>codigo</code>	Código único do filme.

Parameters

<i>unidadesDisponiveis</i>	Número de unidades disponíveis para locação.
<i>_titulo</i>	Título do Blu-ray.
<i>_codigo</i>	Código único do Blu-ray.
<i>_unidadesDisponiveis</i>	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

<i>std::invalid_argument</i>	Se não houver aparelhos disponíveis.
<i>std::invalid_argument</i>	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

<i>dias</i>	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

<i>dias</i>	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

<i>quantidade</i>	Nova quantidade de aparelhos disponíveis.
<i>quantidade</i>	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

<i>_nome</i>	Nome do cliente.
<i>_cpf</i>	CPF do cliente.
<i>_pontos_fidelidade</i>	Pontos de fidelidade acumulados pelo cliente.
<i>_filmes_alugados</i>	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

<i>nome</i>	Nome do cliente.
<i>cpf</i>	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

<i>filme</i>	Filme a ser adicionado.
<i>filme</i>	Ponteiro para o filme a ser adicionado.

6.2.3.2 adicionarPontos()

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

```
int pontos )
```

Adiciona pontos de fidelidade ao cliente.

Parameters

<i>pontos</i>	Pontos a serem adicionados.
<i>pontos</i>	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.

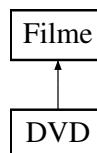
The documentation for this class was generated from the following files:

- 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

<i>categoria</i>	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

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

<i>dias</i>	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

<i>dias</i>	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).

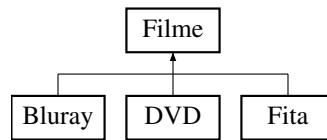
The documentation for this class was generated from the following files:

- 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](#), [BluRay](#), [Fita](#)) e métodos para manipulação desses atributos.

Parameters

<i>titulo</i>	Título do filme.
<i>codigo</i>	Código único do filme.
<i>unidadesDisponiveis</i>	Número de unidades disponíveis para locação.
<i>tipo</i>	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

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

6.4.3.2 ~Filme()

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

Destruitor da classe [Filme](#) (virtual para permitir polimorfismo).

Destruitor 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

<i>quantidade</i>	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

<i>dias</i>	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

<i>dias</i>	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

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

Returns

O objeto de fluxo de saída.

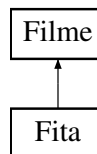
The documentation for this class was generated from the following files:

- [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

<i>rebobinada</i>	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

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

<i>dias</i>	Número de dias pelos quais o filme será alugado.
-------------	--

Returns

O valor total da locação.

Parameters

<i>dias</i>	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.
- vector< [Filme](#) * > [getvectorFilmes](#) ()
Getter para o atributo _catalogo_filmes.
- vector< [Cliente](#) * > [getvectorClientes](#) ()
Getter para o atributo _clientes_cadastrados.

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

<i>cpf</i>	CPF do cliente.
<i>filmes</i>	Vetor de ponteiros para os filmes a serem alugados.
<i>cpf</i>	CPF do cliente que deseja alugar os filmes.
<i>filmes_a_alugar</i>	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

<i>nome</i>	Nome do cliente.
<i>cpf</i>	CPF do cliente.
<i>nome</i>	Nome do cliente a ser cadastrado.
<i>cpf</i>	CPF do cliente a ser cadastrado.

6.6.2.3 cadastrarFilme()

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

Cadastra um novo filme no catálogo.

Parameters

<i>filme</i>	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

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

6.6.2.5 emiteReciboAluguel()

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

Emita o recibo de aluguel para um cliente.

Emita um recibo de aluguel para o cliente.

Parameters

<i>cliente</i>	Ponteiro para o cliente.
<i>filmes</i>	Vetor de ponteiros para os filmes alugados.
<i>cliente</i>	Ponteiro para o cliente que alugou os filmes.
<i>filmes</i>	Vetor de ponteiros para os filmes alugados.

6.6.2.6 emiteReciboDevolucao()

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

Emita o recibo de devolução para um cliente.

Emita um recibo de devolução para o cliente.

Parameters

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

<i>cpf</i>	CPF do cliente.
------------	-----------------

Returns

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

Parameters

<i>cpf</i>	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

<i>codigo_filme</i>	Código do filme.
---------------------	------------------

Returns

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

Parameters

<i>codigo_filme</i>	Código do filme a ser encontrado.
---------------------	-----------------------------------

Returns

Ponteiro para o [Filme](#) encontrado ou nullptr se não for encontrado.

6.6.2.9 getvectorClientes()

```
vector< Cliente * > Locacao::getvectorClientes ( )
```

Getter para o atributo `_clientes_cadastrados`.

Retorna o vetor de clientes de locacao.

Returns

Vetor de clientes.

6.6.2.10 getvectorFilmes()

```
vector< Filme * > Locacao::getvectorFilmes ( )
```

Getter para o atributo `_catalogo_filmes`.

Retorna o vetor de filmes de locacao.

Returns

Vetor de filmes ([DVD](#), [BluRay](#), [Fita](#)).

6.6.2.11 listarClientesCodigo()

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

Lista os clientes cadastrados no sistema ordenados por código.

Lista os clientes ordenados por código.

6.6.2.12 listarClientesNome()

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

Lista os clientes cadastrados no sistema ordenados por nome.

Lista os clientes ordenados por nome.

6.6.2.13 listarFilmesCodigo()

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

Lista os filmes cadastrados no sistema ordenados por código.

Lista os filmes ordenados por código.

6.6.2.14 listarFilmesTitulo()

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

Lista os filmes cadastrados no sistema ordenados por título.

Lista os filmes ordenados por título.

6.6.2.15 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

<i>cpf</i>	CPF do cliente a ser removido.
<i>cpf</i>	CPF do cliente a ser removido.

Exceptions

<i>std::invalid_argument</i>	se o cliente não for encontrado.
------------------------------	----------------------------------

6.6.2.16 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

<i>codigo</i>	Código do filme a ser removido.
<i>codigo</i>	Código do filme a ser removido.

Exceptions

<i>std::invalid_argument</i>	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

- 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 Member Function Documentation

6.7.2.1 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

<i>lista_input</i>	Lista contendo os inputs da operação selecionada.
--------------------	---

Exceptions

<i>std::invalid_argument</i>	se o cliente não for encontrado ou o código do filme for inexistente.
------------------------------	---

Parameters

<i>lista_input</i>	lista de inputs a serem utilizados pela função
--------------------	--

6.7.2.2 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

<i>lista_input</i>	Lista contendo os inputs da operação selecionada.
--------------------	---

Exceptions

<i>std::invalid_argument</i>	se o CPF fornecido for inválido.
------------------------------	----------------------------------

Parameters

<i>lista_input</i>	lista de inputs a serem utilizados pela função
--------------------	--

6.7.2.3 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

<i>lista_input</i>	Lista contendo os inputs da operação selecionada.
--------------------	---

Exceptions

<i>std::invalid_argument</i>	se o tipo de filme for inválido.
------------------------------	----------------------------------

Parameters

<i>lista_input</i>	lista de inputs a serem utilizados pela função
--------------------	--

6.7.2.4 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

<i>lista_input</i>	Lista contendo os inputs da operação selecionada.
--------------------	---

Exceptions

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

Parameters

<i>lista_input</i>	lista de inputs a serem utilizados pela função
--------------------	--

6.7.2.5 displayHelp()

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

Exibe informações de ajuda sobre os comandos disponíveis no sistema.

Exibe o menu de ajuda com os comandos disponíveis.

6.7.2.6 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

<i>lista_input</i>	Lista contendo os inputs da operação selecionada.
--------------------	---

Exceptions

<i>std::invalid_argument</i>	se o comando não existir.
------------------------------	---------------------------

Parameters

<i>lista_input</i>	lista de inputs a serem utilizados pela função
--------------------	--

6.7.2.7 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

<i>nome_arquivo</i>	Nome do arquivo a ser lido.
---------------------	-----------------------------

Exceptions

<i>std::invalid_argument</i>	se o arquivo não existir.
------------------------------	---------------------------

Parameters

<i>nome_arquivo</i>	string com o nome do arquivo a ser aberto para leitura no sistema
---------------------	---

6.7.2.8 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

<i>lista_input</i>	Lista contendo os inputs da operação selecionada.
--------------------	---

Exceptions

<i>std::invalid_argument</i>	se a opção de listagem for inválida.
------------------------------	--------------------------------------

Parameters

<i>lista_input</i>	lista de inputs a serem utilizados pela função
--------------------	--

6.7.2.9 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

<i>lista_input</i>	Lista contendo os inputs da operação selecionada.
--------------------	---

Exceptions

<i>std::invalid_argument</i>	se a opção de listagem for inválida.
------------------------------	--------------------------------------

Parameters

<i>lista_input</i>	lista de inputs a serem utilizados pela função
--------------------	--

6.7.2.10 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

<i>input</i>	String contendo o input do usuario.
<i>input</i>	string com o input inserido pelo usuário.

6.7.2.11 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

<i>lista_input</i>	Lista contendo os inputs da operação selecionada.
--------------------	---

Exceptions

<i>std::out_of_range</i>	se nenhum filme estiver alugado para o cliente.
--------------------------	---

Parameters

<i>lista_input</i>	lista de inputs a serem utilizados pela função
--------------------	--

6.7.2.12 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

<i>lista_input</i>	Lista contendo os inputs da operação selecionada.
--------------------	---

Exceptions

<code>std::invalid_argument</code>	se nenhum filme com os códigos fornecidos for encontrado.
------------------------------------	---

Parameters

<code>lista_input</code>	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

<i>os</i>	Objeto de fluxo de saída.
<i>tipo</i>	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     Tipo tipo;
00036     int unidadesDisponiveis;
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

<i>input</i>	A string a ser convertida.
--------------	----------------------------

Returns

A string com a primeira letra em maiúscula.

Converte a primeira letra de uma string para maiúscula.

Parameters

<i>input</i>	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

<i>input</i>	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

<i>input</i>	A string a ser convertida.
--------------	----------------------------

Returns

A string convertida para letras minúsculas.

Converte uma string para letras minúsculas.

Parameters

<i>input</i>	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

<i>input</i>	A string a ser convertida.
--------------	----------------------------

Returns

A string convertida para letras maiúsculas.

Converte uma string para letras maiúsculas.

Parameters

<i>input</i>	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
00129     vector<Filme*> getvectorFilmes();
00130
00135     vector<Cliente*> getvectorClientes();
00136 };
00137
00138 #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:
00026     void displayHelp();
00027
00032     void cadastrarFilme(std::deque<std::string> lista_input);
00033
00038     void cadastrarCliente(std::deque<std::string> lista_input);
00039
00044     void removerCliente(std::deque<std::string> lista_input);
00045
00050     void removerFilme(std::deque<std::string> lista_input);
00051
00056     void listarFilmes(std::deque<std::string> lista_input);
00057
00062     void listarClientes(std::deque<std::string> lista_input);
00063
00068     void devolverFilmes(std::deque<std::string> lista_input);
00069
00074     void alugarFilmes(std::deque<std::string> lista_input);
00075
00080     void lerArquivo(string nome_arquivo);
00081
00086     bool gerenciaOpcoes(string comando, std::deque<std::string> lista_input);
00087
00093     std::tuple<std::string, std::deque<std::string>> processaInput(std::string input);
00094 };
00095
00096 #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

<i>input</i>	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

<i>input</i>	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

<i>input</i>	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

<i>input</i>	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"
#include "fita.hpp"
#include "bluray.hpp"
#include "cliente.hpp"
```

Macros

- [#define DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN](#)

Functions

- [TEST_CASE](#) ("Testando gets")
- [TEST_CASE](#) ("Testando vetor de filmes")
- [TEST_CASE](#) ("Testando vetor de clientes")
- [TEST_CASE](#) ("Aluguel de filmes")

7.29.1 Macro Definition Documentation

7.29.1.1 DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN

```
#define DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN
```

7.29.2 Function Documentation

7.29.2.1 TEST_CASE() [1/4]

```
TEST_CASE (
    "Aluguel de filmes" )
```

7.29.2.2 TEST_CASE() [2/4]

```
TEST_CASE (
    "Testando gets" )
```

7.29.2.3 TEST_CASE() [3/4]

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

7.29.2.4 TEST_CASE() [4/4]

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

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" )
```

