PROJETO FINAL - PDS2

Índice Hierárquico

Hierarquia de Classes

Esta lista de hierarquias está parcialmente ordenada (ordem alfabética):

Índice dos Componentes

Lista de Classes

Aqui estão as classes, estruturas, uniões e interfaces e suas respectivas descrições:

Índice dos Arquivos

Lista de Arquivos

Esta é a lista de todos os arquivos documentados e suas respectivas descrições:

Classes

Referência da Classe CadastroJogadores

Classe que cuida do cadastro de jogadores.
`#include <Cadastro.hpp>`

Membros Públicos

- std::string cadastrar (const std::string &apelido, const std::string &nome)
  Método para cadastrar novos jogadores e adicionar ao mapa.

- std::string remover (const std::string &apelido)
  Método para remover cadastro de jogadores.

- void listar (char ordenarPor)
  Método para listar os jogadores cadastrados.

- void **salvarEmArquivo** ()
  Salvar os dados dos jogadores cadastrados em um arquivo de texto.

- void **carregarDoArquivo** ()
  Carregar os dados de jogadores salvos em um arquivo de texto.

- bool jogadorExiste (const std::string &apelido) const
  Verifica se existe um cadastro.

- Jogador & getJogador (const std::string &apelido)
  Retorna um objeto do tipo jogador.

Descrição detalhada

Classe que cuida do cadastro de jogadores.

Documentação das funções

std::string CadastroJogadores::cadastrar (const std::string & apelido, const std::string & nome)

Método para cadastrar novos jogadores e adicionar ao mapa.

Parâmetros

apelido	Apelido único para cadastro.
nome	Nome do jogador.

Jogador & CadastroJogadores::getJogador (const std::string & apelido)

Retorna um objeto do tipo jogador.

Parâmetros

apelido	Apelido do jogador que deve ser retornado.

bool CadastroJogadores::jogadorExiste (const std::string & apelido) const

Verifica se existe um cadastro.

Parâmetros

apelido	Apelido que será verificado.

void CadastroJogadores::listar (char ordenarPor)

Método para listar os jogadores cadastrados.

Parâmetros

ordernarPor	Representa como a listagem será feita, ou seja, por apelido ("A") ou por ranking de vitórias ("R").

std::string CadastroJogadores::remover (const std::string & apelido)

Método para remover cadastro de jogadores.

Parâmetros

apelido	Apelido do jogador que será removido.

A documentação para essa classe foi gerada a partir do seguinte arquivo:

- Cadastro.hpp

Referência da Estrutura Jogador::Estatisticas

Número de vitórias e derrotas em cada jogo.

`#include <jogador.hpp>`

Atributos Públicos

- int **vitorias_reversi** = 0
- int **derrotas_reversi** = 0
- int **vitorias_lig4** = 0
- int **derrotas_lig4** = 0
- int **vitorias_velha** = 0
- int **derrotas_velha** = 0

Descrição detalhada

Número de vitórias e derrotas em cada jogo.

A documentação para essa estrutura foi gerada a partir do seguinte arquivo:

- jogador.hpp

Referência da Classe Jogador

Classe que representa um jogador, com seus dados e estatísticas.

`#include <jogador.hpp>`

Componentes

struct Estatisticas*Número de vitórias e derrotas em cada jogo.*

Atributos Públicos

- std::string **apelido**
- std::string **nome**
- Estatisticas **stats**

Descrição detalhada

Classe que representa um jogador, com seus dados e estatísticas.

A documentação para essa classe foi gerada a partir do seguinte arquivo:

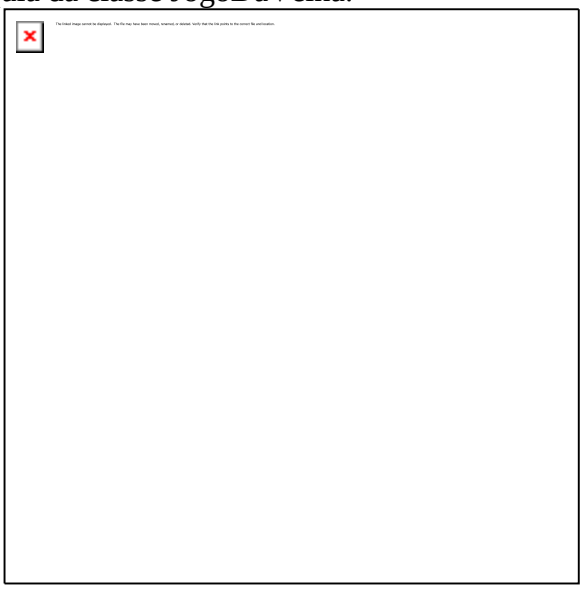
- jogador.hpp

Referência da Classe JogoDaVelha

Classe que implementa o jogo da velha, especializando as funções da classe abstrata.

`#include <JogoDaVelha.hpp>`

Diagrama de hierarquia da classe JogoDaVelha:



Membros Públicos

- void DefinirApelidos (const std::string &x, const std::string &o)
  Assinala "X" e "O" para os respectivos jogadores.

- bool JogadaValida (int linha, int coluna) const override
  Verifica se a jogada escolhida é válida.

- void Reiniciar () override
  Reinicia o tabuleiro, deixando-o vazio novamente.

- bool VerificarVitoria (char) const override
  Verifica se algum dos jogadores ganhou após sua última jogada.

- void RealizarJogada (int, int, char) override
  Realiza uma jogada conforme escolhido.

- int ExecutarPartida () override
  Implementa o loop de gameplay do jogo.

- int Jogar () override
  Chama o método para executar a partida.

- void ImprimirTabuleiro () const override
  Imprime o tabuleiro completo.

Membros Públicos herdados de JogoDeTabuleiro

- JogoDeTabuleiro (int linhas, int colunas)
  Construtor padrão do tabuleiro, inicia todas as células como vazias.

- virtual **~JogoDeTabuleiro** ()=default
  Destrutor virtual do tabuleiro.

Outros membros herdados

Atributos Protegidos herdados de JogoDeTabuleiro

- std::vector< std::vector< char > > **tabuleiro**
  Estrutura de dados vector para simular o tabuleiro representado por caracteres.

- int **linhas**
- int colunas

Descrição detalhada

Classe que implementa o jogo da velha, especializando as funções da classe abstrata.

Documentação das funções

void JogoDaVelha::DefinirApelidos (const std::string & x, const std::string & o)

Assinala "X" e "O" para os respectivos jogadores.

Parâmetros

x	O apelido do jogador que terá "X" como símbolo.
o	O apelido do jogador que terá "O" como símbolo.

int JogoDaVelha::ExecutarPartida () `[override]`, `[virtual]`

Implementa o loop de gameplay do jogo.

Retorna

Número que indica o jogador vencedor.

Implementa JogoDeTabuleiro.

void JogoDaVelha::ImprimirTabuleiro () const `[override]`, `[virtual]`

Imprime o tabuleiro completo.

Implementa JogoDeTabuleiro.

bool JogoDaVelha::JogadaValida (int linha, int coluna) const `[override]`, `[virtual]`

Verifica se a jogada escolhida é válida.

Parâmetros

linha	Linha da jogada.
coluna	Coluna da jogada.

Retorna

Valor que indica a validade da jogada (true ou false).

Implementa JogoDeTabuleiro.

int JogoDaVelha::Jogar () `[override], [virtual]`

Chama o método para executar a partida.

Retorna

Número que indica o jogador vencedor.

Implementa JogoDeTabuleiro.

void JogoDaVelha::RealizarJogada (int linha, int coluna, char jogador) `[override], [virtual]`

Realiza uma jogada conforme escolhido.

Parâmetros

linha	Linha da jogada.
coluna	Coluna da jogada.
jogador	Jogador 1 (X) ou jogador 2 (O).

Implementa JogoDeTabuleiro.

void JogoDaVelha::Reiniciar () `[override], [virtual]`

Reinicia o tabuleiro, deixando-o vazio novamente.

Implementa JogoDeTabuleiro.

bool JogoDaVelha::VerificarVitoria (char jogador) const `[override], [virtual]`

Verifica se algum dos jogadores ganhou após sua última jogada.

Parâmetros

jogador	Jogador 1 (X) ou jogador 2 (O).

Retorna

True caso o jogador tenha ganhado, false caso contrário.

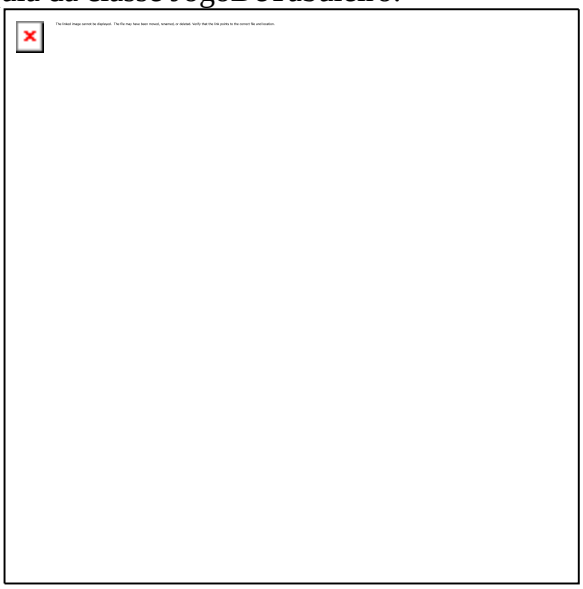
Implementa JogoDeTabuleiro.

A documentação para essa classe foi gerada a partir do seguinte arquivo:

- JogoDaVelha.hpp

Referência da Classe JogoDeTabuleiro

Classe genérica com funções comuns para todos os jogos de tabuleiro.
`#include <JogoDeTabuleiro.hpp>`
Diagrama de hierarquia da classe JogoDeTabuleiro:



Membros Públicos

- JogoDeTabuleiro (int linhas, int colunas)
  Construtor padrão do tabuleiro, inicia todas as células como vazias.

- virtual **~JogoDeTabuleiro** ()=default
  Destrutor virtual do tabuleiro.

- virtual int Jogar ()=0
  Chama o método para executar a partida.

- virtual void ImprimirTabuleiro () const =0
  Imprime o tabuleiro completo.

- virtual bool JogadaValida (int linha, int coluna) const =0
  Verifica se a jogada escolhida é válida.

- virtual void RealizarJogada (int linha, int coluna, char jogador)=0
  Realiza uma jogada conforme escolhido.

- virtual void Reiniciar ()=0
  Reinicia o tabuleiro, deixando-o vazio novamente.

- virtual bool VerificarVitoria (char jogador) const =0
  Verifica se algum dos jogadores ganhou após sua última jogada.

- virtual int ExecutarPartida ()=0

Implementa o loop de gameplay do jogo.

Atributos Protegidos

- std::vector< std::vector< char > > **tabuleiro**
  Estrutura de dados vector para simular o tabuleiro representado por caracteres.

- int **linhas**
- int colunas

Descrição detalhada

Classe genérica com funções comuns para todos os jogos de tabuleiro.

Construtores e Destrutores

JogoDeTabuleiro::JogoDeTabuleiro (int linhas, int colunas) `[inline]`

Construtor padrão do tabuleiro, inicia todas as células como vazias.

<Número de colunas no tabuleiro.

Parâmetros

linhas	Quantidade de linhas a ser instanciada.
colunas	Quantidade de colunas a ser instanciada.

Documentação das funções

virtual int JogoDeTabuleiro::ExecutarPartida () `[pure virtual]`

Implementa o loop de gameplay do jogo.

Retorna

Número que indica o jogador vencedor.

Implementado por JogoDaVelha, Lig4 e Reversi.

virtual void JogoDeTabuleiro::ImprimirTabuleiro () const `[pure virtual]`

Imprime o tabuleiro completo.

Implementado por JogoDaVelha, Lig4 e Reversi.

virtual bool JogoDeTabuleiro::JogadaValida (int linha, int coluna) const `[pure virtual]`

Verifica se a jogada escolhida é válida.

Parâmetros

linha	Linha da jogada.
coluna	Coluna da jogada.

Retorna

Valor que indica a validade da jogada (true ou false).

Implementado por JogoDaVelha, Lig4 e Reversi.

virtual int JogoDeTabuleiro::Jogar () [pure virtual]

Chama o método para executar a partida.

Retorna

Número que indica o jogador vencedor.

Implementado por JogoDaVelha, Lig4 e Reversi.

virtual void JogoDeTabuleiro::RealizarJogada (int linha, int coluna, char jogador) [pure virtual]

Realiza uma jogada conforme escolhido.

Parâmetros

linha	Linha da jogada.
coluna	Coluna da jogada.
jogador	Jogador 1 (X) ou jogador 2 (O).

Implementado por JogoDaVelha, Lig4 e Reversi.

virtual void JogoDeTabuleiro::Reiniciar () [pure virtual]

Reinicia o tabuleiro, deixando-o vazio novamente.

Implementado por JogoDaVelha, Lig4 e Reversi.

virtual bool JogoDeTabuleiro::VerificarVitoria (char jogador) const [pure virtual]

Verifica se algum dos jogadores ganhou após sua última jogada.

Parâmetros

jogador	Jogador 1 (X) ou jogador 2 (O).

Retorna

True caso o jogador tenha ganhado, false caso contrário.

Implementado por JogoDaVelha, Lig4 e Reversi.

Atributos

int JogoDeTabuleiro::colunas [protected]

<Número de linhas no tabuleiro.

A documentação para essa classe foi gerada a partir do seguinte arquivo:

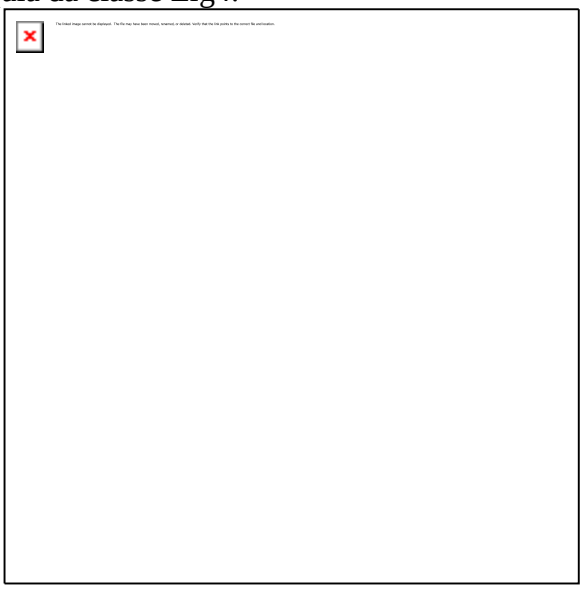
- JogoDeTabuleiro.hpp

Referência da Classe Lig4

Classe que herda a classe JogoDeTabuleiro e implementa lógica do jogo Lig4.

```
#include <Lig4.hpp>
```

Diagrama de hierarquia da classe Lig4:



Membros Públicos

- void ImprimirTabuleiro () const override
  Imprime o tabuleiro completo.

- bool JogadaValida (int linha, int coluna) const override
  Verifica se a jogada escolhida é válida.

- void RealizarJogada (int linha, int coluna, char jogador) override
  Realiza uma jogada conforme escolhido.

- void Reiniciar () override
  Reinicia o tabuleiro, deixando-o vazio novamente.

- bool VerificarVitoria (char jogador) const override
  Verifica se algum dos jogadores ganhou após sua última jogada.

- virtual int Jogar () override
  Chama o método para executar a partida.

- int ExecutarPartida () override
  Implementa o loop de gameplay do jogo.

Membros Públicos herdados de JogoDeTabuleiro

- JogoDeTabuleiro (int linhas, int colunas)
  Construtor padrão do tabuleiro, inicia todas as células como vazias.

- virtual **~JogoDeTabuleiro** ()=default
  Destrutor virtual do tabuleiro.

Outros membros herdados

Atributos Protegidos herdados de JogoDeTabuleiro

- std::vector< std::vector< char > > **tabuleiro**
  Estrutura de dados vector para simular o tabuleiro representado por caracteres.

- int **linhas**
- int colunas

Descrição detalhada

Classe que herda a classe JogoDeTabuleiro e implementa lógica do jogo Lig4.

Documentação das funções

int Lig4::ExecutarPartida () `[override]`, `[virtual]`

Implementa o loop de gameplay do jogo.

Retorna

Número que indica o jogador vencedor.

Implementa JogoDeTabuleiro.

void Lig4::ImprimirTabuleiro () const `[override]`, `[virtual]`

Imprime o tabuleiro completo.

Implementa JogoDeTabuleiro.

bool Lig4::JogadaValida (int linha, int coluna) const `[override]`, `[virtual]`

Verifica se a jogada escolhida é válida.

Parâmetros

linha	Linha da jogada.
coluna	Coluna da jogada.

Retorna

Valor que indica a validade da jogada (true ou false).

Implementa JogoDeTabuleiro.

virtual int Lig4::Jogar () `[override]`, `[virtual]`

Chama o método para executar a partida.

Retorna

Número que indica o jogador vencedor.

Implementa JogoDeTabuleiro.

void Lig4::RealizarJogada (int linha, int coluna, char jogador) [override], [virtual]

Realiza uma jogada conforme escolhido.

Parâmetros

linha	Linha da jogada.
coluna	Coluna da jogada.
jogador	Jogador 1 (X) ou jogador 2 (O).

Implementa JogoDeTabuleiro.

void Lig4::Reiniciar () [override], [virtual]

Reinicia o tabuleiro, deixando-o vazio novamente.

Implementa JogoDeTabuleiro.

bool Lig4::VerificarVitoria (char jogador) const [override], [virtual]

Verifica se algum dos jogadores ganhou após sua última jogada.

Parâmetros

jogador	Jogador 1 (X) ou jogador 2 (O).

Retorna

True caso o jogador tenha ganhado, false caso contrário.

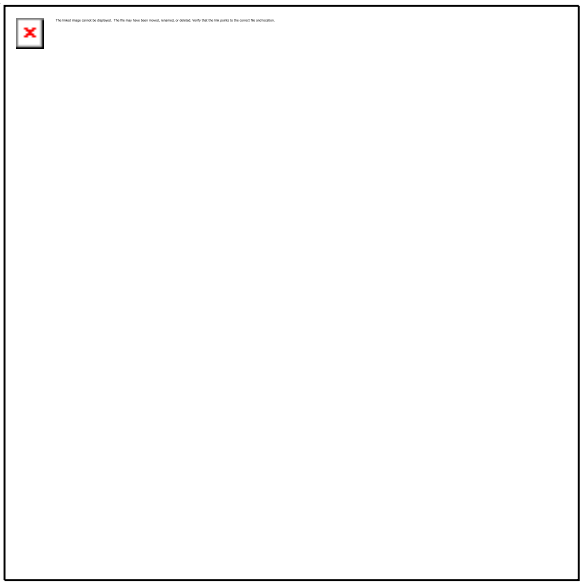
Implementa JogoDeTabuleiro.

A documentação para essa classe foi gerada a partir do seguinte arquivo:

- Lig4.hpp

Referência da Classe Reversi

Diagrama de hierarquia da classe Reversi:



Membros Públicos

- void **DefinirApelidos** (const std::string &x, const std::string &o)
- std::string ObterApelido (char simbolo) const
  Obtém o apelido de um jogador.

- virtual int Jogar () override
  Chama o método para executar a partida.

- virtual void ImprimirTabuleiro () const override
  Imprime o tabuleiro completo.

- virtual bool JogadaValida (int linha, int coluna) const override
  Verifica se a jogada escolhida é válida.

- virtual void RealizarJogada (int linha, int coluna, char jogador) override
  Realiza uma jogada conforme escolhido.

- virtual void Reiniciar () override
  Reinicia o tabuleiro, deixando-o vazio novamente.

- virtual bool VerificarVitoria (char jogador) const override
  Verifica se algum dos jogadores ganhou após sua última jogada.

- virtual int ExecutarPartida () override
  Implementa o loop de gameplay do jogo.

Membros Públicos herdados de JogoDeTabuleiro

- JogoDeTabuleiro (int linhas, int colunas)

Construtor padrão do tabuleiro, inicia todas as células como vazias.

- virtual **~JogoDeTabuleiro** ()=default
  Destrutor virtual do tabuleiro.

Outros membros herdados

Atributos Protegidos herdados de JogoDeTabuleiro

- std::vector< std::vector< char > > **tabuleiro**
  Estrutura de dados vector para simular o tabuleiro representado por caracteres.

- int **linhas**
- int colunas

Documentação das funções

virtual int Reversi::ExecutarPartida () `[override]`, `[virtual]`

Implementa o loop de gameplay do jogo.

Retorna

Número que indica o jogador vencedor.

Implementa JogoDeTabuleiro.

virtual void Reversi::ImprimirTabuleiro () const `[override]`, `[virtual]`

Imprime o tabuleiro completo.

Implementa JogoDeTabuleiro.

virtual bool Reversi::JogadaValida (int linha, int coluna) const `[override]`, `[virtual]`

Verifica se a jogada escolhida é válida.

Parâmetros

linha	Linha da jogada.
coluna	Coluna da jogada.

Retorna

Valor que indica a validade da jogada (true ou false).

Implementa JogoDeTabuleiro.

virtual int Reversi::Jogar () `[override]`, `[virtual]`

Chama o método para executar a partida.

Retorna

Número que indica o jogador vencedor.

Implementa JogoDeTabuleiro.

std::string Reversi::ObterApelido (char simbolo) const

Obtém o apelido de um jogador.

Parâmetros

simbolo	"X" ou "O" para representar cada jogador.

Retorna

O apelido em si.

virtual void Reversi::RealizarJogada (int linha, int coluna, char jogador) [override], [virtual]

Realiza uma jogada conforme escolhido.

Parâmetros

linha	Linha da jogada.
coluna	Coluna da jogada.
jogador	Jogador 1 (X) ou jogador 2 (O).

Implementa JogoDeTabuleiro.

virtual void Reversi::Reiniciar () [override], [virtual]

Reinicia o tabuleiro, deixando-o vazio novamente.

Implementa JogoDeTabuleiro.

virtual bool Reversi::VerificarVitoria (char jogador) const [override], [virtual]

Verifica se algum dos jogadores ganhou após sua última jogada.

Parâmetros

jogador	Jogador 1 (X) ou jogador 2 (O).

Retorna

True caso o jogador tenha ganhado, false caso contrário.

Implementa JogoDeTabuleiro.

A documentação para essa classe foi gerada a partir do seguinte arquivo:

- Reversi.hpp

Arquivos

Cadastro.hpp

```
1 #ifndef CADASTRO_HPP
2 #define CADASTRO_HPP
3 #pragma once
4
5 #include <map>
6 #include <string>
7 #include <vector>
8 #include "Jogador.hpp"
9
13 class CadastroJogadores
14 {
15     private:
19         std::map<std::string, Jogador> jogadores;
23         const std::string arquivo = "jogadores.txt";
24
25     public:
26         // Operações básicas
32         std::string cadastrar(const std::string& apelido, const std::string& nome);
37         std::string remover(const std::string& apelido);
42         void listar(char ordenarPor);
43
44         // Persistência
48         void salvarEmArquivo();
52         void carregarDoArquivo();
53
54         // Consultas
59         bool jogadorExiste(const std::string& apelido) const;
64         Jogador& getJogador(const std::string& apelido);
65 };
66
67 #endif
```

jogador.hpp

```
1 #ifndef JOGADOR_HPP
2 #define JOGADOR_HPP
3
4 #include <string>
5 #include <map>
9 class Jogador
10 {
11 public:
12     std::string apelido;
13     std::string nome;
17     struct Estatisticas
18     {
19         int vitorias_reversi = 0;
20         int derrotas_reversi = 0;
21         int vitorias lig4 = 0;
22         int derrotas lig4 = 0;
23         int vitorias_velha = 0;
24         int derrotas_velha = 0;
25     };
26     Estatisticas stats;
27 };
28
29 #endif
```

JogoDaVelha.hpp

```
1 #ifndef JOGO_DA_VELHA_H
2 #define JOGO_DA_VELHA_H
3
4 #include "JogoDeTabuleiro.hpp"
8 class JogoDaVelha: public JogoDeTabuleiro{
9
10 private:
11
12
13     std::string jogadorX;
14     std::string jogadorO;
20     std::string ObterApelido(char simbolo) const;
21
22
23 public:
24
30     void DefinirApelidos(const std::string& x, const std::string& o);
31     JogoDaVelha();
32     virtual ~JogoDaVelha() = default;
33     bool JogadaValida(int linha, int coluna) const override;
34     void Reiniciar() override;
35     bool VerificarVitoria(char) const override;
36     void RealizarJogada(int, int , char) override;
37     int ExecutarPartida() override;
38     int Jogar() override;
39     void ImprimirTabuleiro() const override;
40 };
41 #endif
```

JogoDeTabuleiro.hpp

```
1 #ifndef JOGODETABULEIRO_HPP
2 #define JOGODETABULEIRO_HPP
3
4 #include <vector>
5 #include <iostream>
6
10 class JogoDeTabuleiro
11 {
12     protected:
16         std::vector<std::vector<char>> tabuleiro;
17         int linhas;
18         int colunas;
19
20     public:
26         JogoDeTabuleiro(int linhas, int colunas) : linhas(linhas), colunas(colunas),
tabuleiro(linhas, std::vector<char>(colunas, '.')) {}
27
31         virtual ~JogoDeTabuleiro() = default;
32
33         // Métodos puramente virtuais (obrigatórios para classes derivadas):
34
39         virtual int Jogar() = 0;
40
44         virtual void ImprimirTabuleiro() const = 0;
45
52         virtual bool JogadaValida(int linha, int coluna) const = 0;
53
60         virtual void RealizarJogada(int linha, int coluna, char jogador) = 0;
61
65         virtual void Reiniciar() = 0;
66
72         virtual bool VerificarVitoria(char jogador) const = 0;
73
78         virtual int ExecutarPartida() = 0;
79 };
80
81 #endif
```

Lig4.hpp

```
1 #ifndef LIG4_HPP
2 #define LIG4_HPP
3
4 #include "JogoDeTabuleiro.hpp"
5
9 class Lig4 : public JogoDeTabuleiro
10 {
11     public:
12     Lig4();
13
14     virtual ~Lig4();
15
16     void ImprimirTabuleiro() const override;
17
18     bool JogadaValida(int linha, int coluna) const override;
19
20     void RealizarJogada(int linha, int coluna, char jogador) override;
21
22     void Reiniciar() override;
23
24     bool VerificarVitoria(char jogador) const override;
25
26     virtual int Jogar() override;
27
28     int ExecutarPartida() override;
29
30     private:
39     bool VerificarDirecao(int linha, int coluna, int dLinha, int dColuna, char jogador)
const;
40
41     std::string jogadorX;
42     std::string jogadorO;
43
49     void DefinirApelidos(const std::string& x, const std::string& o);
50
56     std::string ObterApelido(char simbolo) const;
57
58 };
59
60 #endif
```

Reversi.hpp

```
#ifndef REVERSI_HPP
#define REVERSI_HPP

#include "JogoDeTabuleiro.hpp"
#include <vector>
#include <string>

class Reversi : public JogoDeTabuleiro {
private:
    char jogadorAtual;
    bool jogoFinalizado;
    std::string jogadorX;
    std::string jogadorO;

    // Métodos auxiliares
    bool VerificarDirecao(int linha, int coluna, int dirLinha, int dirColuna) const;
    void VirarDiscos(int linha, int coluna, int dirLinha, int dirColuna);
    bool PodeJogar(char jogador) const;
    bool ConverterEntrada(const std::string& entrada, int& linha, int& coluna) const;
    void ExibirInstrucoes() const;
public:
    Reversi();
    virtual ~Reversi() override = default;

    void DefinirApelidos(const std::string& x, const std::string& o);
    std::string ObterApelido(char simbolo) const;
    // Métodos sobrescritos
    virtual int Jogar() override;
    virtual void ImprimirTabuleiro() const override;
    virtual bool JogadaValida(int linha, int coluna) const override;
    virtual void RealizarJogada(int linha, int coluna, char jogador) override;
    virtual void Reiniciar() override;
    virtual bool VerificarVitoria(char jogador) const override;
    virtual int ExecutarPartida() override;
};

#endif
```

Sumário

INDEX