



INSTITUTO POLITÉCNICO DE COIMBRA
INSTITUTO SUPERIOR DE ENGENHARIA DE COIMBRA

Curso:
Engenharia Informática

Ano letivo:
2020/2021

2º Ano 2º Semestre
Programação Avançada

Relatório

Trabalho Prático - Meta 2

Trabalho realizado por:
Nome: **Daniel Moreira Ribeiro**
Número: **2017013425**
Turma: **P3**

Coimbra, Junho 2021

Introdução	3
Diagrama da máquina de estados	4
Imagem do diagrama da máquina de estados	4
Estados	5
AwaitsStart	5
AwaitsGameMode	5
AwaitsPlayerName	5
AwaitsTheNextMove	6
AwaitsColumnChoice	6
AwaitsMiniGame	6
AwaitsWritingMiniGame	7
AwaitsMathMiniGame	7
AwaitsExit	7
AwaitsReplayID	7
AwaitsNextReplayMove	8
Descrição das classes	9
Descrição do relacionamento entre classes	12
Decisões tomadas na implementação	13
Funcionalidades Implementadas	14
Conclusão	15

Introdução

Este trabalho pretendia que fosse implementado uma versão adaptado do jogo 4 em linha. Os alunos deveriam usar os padrões apresentados nas aulas, nomeadamente o padrão da máquina de estados para implementar este programa.

O trabalho é dividido em 2 fases, sendo que esta 1ª fase se debruça na parte lógica do jogo acompanhada com uma interface de texto.

A segunda meta tinha como objetivo a implementação do jogo em JAVAFX, possibilitando assim jogar numa interface gráfica.

Estados

AwaitsStart

O estado “*AwaitsStart*” é o primeiro estado a ser chamado quando se inicia a aplicação. Tem a função de menu inicial.

O utilizador pode efetuar as seguintes escolhas:

- **Encerrar a aplicação;**
- **Visitar o menu de escolha de replays** - Ir para o estado “*AwaitsReplayID*” através da função “*watchReplay()*”;
- **Iniciar um novo jogo** - Ir para o estado “*AwaitsGameMode*” através da função “*start()*”;
- **Carregar um jogo;**

AwaitsGameMode

O estado “*AwaitsGameMode*” é responsável pela seleção do modo de jogo.

O utilizador pode efetuar as seguintes escolhas:

- **Voltar atrás ao Menu Inicial** - Ir para o estado “*AwaitsStart*” através da função “*goBack()*”;
- **Escolher jogar o jogo das seguintes formas:**
 - **CPU vs CPU** - Ir para o estado “*AwaitsPlayerName*” através da função “*chooseGameMode(int numberOfHumanPlayers)*” sendo que o valor da variável “*numberOfHumanPlayers*” é obrigatoriamente “0”;
 - **Humano vs CPU** - Ir para o estado “*AwaitsPlayerName*” através da função “*chooseGameMode(int numberOfHumanPlayers)*” sendo que o valor da variável “*numberOfHumanPlayers*” é obrigatoriamente “1”;
 - **Humano vs Humano** - Ir para o estado “*AwaitsPlayerName*” através da função “*chooseGameMode(int numberOfHumanPlayers)*” sendo que o valor da variável “*numberOfHumanPlayers*” é obrigatoriamente “2”;

AwaitsPlayerName

O estado “*AwaitsPlayerName*” é responsável pela atribuição dos nomes aos jogadores.

O utilizador pode efetuar as seguintes escolhas:

- **Retornar à seleção do modo de jogo caso se esteja a escolher o nome do primeiro jogador** - Ir para o estado “*AwaitsGameMode*” através da função “*goBack()*”;
- **Retornar à seleção do nome do primeiro jogador caso se esteja a escolher o nome do segundo jogador** - Ir para o estado “*AwaitsPlayerName*” através da função “*goBack()*”;
- **Escolher o nome do primeiro jogador caso seja a sua vez** - Ir para o estado “*AwaitsPlayerName*” através da função “*choosePlayerName(String playerName)*”;
- **Escolher o nome do segundo jogador caso seja a sua vez** - Ir para o estado “*AwaitsTheNextMove*” através da função “*choosePlayerName(String playerName)*”;

AwaitsTheNextMove

O estado *"AwaitsTheNextMove"* é responsável pela escolha do tipo de peça a jogar na jogada em questão.

O utilizador pode efetuar as seguintes escolhas:

- **Retornar ao menu inicial** - Ir para o estado *"AwaitsStart"* através da função *"goBack()"*;
- **Jogar uma peça normal** - Ir para o estado *"AwaitsColumnChoice"* através da função *"chooseGamePiece(boolean isSpecial)"* sendo que o valor da variável *"isSpecial"* é obrigatoriamente *"false"*;
- **Jogar uma peça especial caso possua peças especiais e seja humano** - Ir para o estado *"AwaitsColumnChoice"* através da função *"chooseGamePiece(boolean isSpecial)"* sendo que o valor da variável *"isSpecial"* é obrigatoriamente *"true"*;
- **Retornar ao momento anterior do tabuleiro de jogo caso possua créditos para isso e seja humano** - Continuar no estado *"AwaitsTheNextMove"* através da função *"undoMove()"*;
- **Guardar o jogo;**

AwaitsColumnChoice

O estado *"AwaitsColumnChoice"* é responsável pela escolha da coluna em que a peça atual será jogada.

O utilizador pode efetuar as seguintes escolhas:

- **Retornar à seleção do tipo de peça** - Ir para o estado *"AwaitsTheNextMove"* através da função *"goBack()"*;
- **Escolher a coluna em que se deseja colocar a peça caso o jogador seja humano** - Através da função *"chooseColumn(int column)"* em que a variável *"column"* possui obrigatoriamente um valor entre *"0"* e *"6"*, ir para o estado *"AwaitsExit"* caso a coluna selecionada faça o jogo acabar, ir para o estado *"AwaitsMiniGame"* caso a próxima jogada corresponda a um mini jogo ou ir para o estado *"AwaitsTheNextMove"* caso o jogo ainda não tenha acabado após esta jogada;
- **Simular a escolha da coluna em que se deseja colocar a peça caso o jogador não seja humano** - Através da função *"chooseColumn(int column)"* em que a variável *"column"* possui obrigatoriamente o valor *"7"*, ir para o estado *"AwaitsExit"* caso a coluna selecionada faça o jogo acabar ou ir para o estado *"AwaitsTheNextMove"* caso o jogo ainda não tenha acabado após esta jogada;

AwaitsMiniGame

O estado *"AwaitsMiniGame"* é responsável pela interpretação da vontade do utilizador relativamente à intenção de participar num mini jogo.

O utilizador pode efetuar as seguintes escolhas:

- **Recusar participar no mini jogo** - Ir para o estado *"AwaitsTheNextMove"* através da função *"playMiniGame(boolean decision)"* em que o valor da variável *"decision"* é obrigatoriamente *"false"*;
- **Aceitar participar no mini jogo** - Através da função *"playMiniGame(boolean decision)"* em que o valor da variável *"decision"* é obrigatoriamente *"true"*, ir para o estado *"AwaitsWritingMiniGame"* se o mini jogo anterior do jogador em questão foi o de matemática ou ir para o estado *"AwaitsMathMiniGame"* se o mini jogo anterior do jogador em questão foi o de escrita;

AwaitsWritingMiniGame

O estado *"AwaitsWritingMiniGame"* é responsável pelo mini jogo de escrita.

O utilizador pode efetuar as seguintes escolhas:

- **Responder ao mini jogo** - Através da função *"AnswerTheWritingMiniGame(String answer)"*, ir para o estado *"AwaitsTheNextMove"* caso o mini jogo tenha sido vencido ou não seja a vez do próximo jogador jogar o mini jogo, ou ir para o estado *"AwaitsMiniGame"* caso não tenha vencido o mini jogo e é a vez do próximo jogador jogar o mini jogo;

AwaitsMathMiniGame

O estado *"AwaitsMathMiniGame"* é responsável pelo mini jogo de matemática.

O utilizador pode efetuar as seguintes escolhas:

- **Responder ao mini jogo** - Através da função *"AnswerTheMathMiniGame(int answer)"*, continuar no mesmo estado caso o jogo ainda não tenha acabado, ir para o estado *"AwaitsTheNextMove"* caso o mini jogo tenha sido vencido ou não seja a vez do próximo jogador jogar o mini jogo, ou ir para o estado *"AwaitsMiniGame"* caso não tenha vencido o mini jogo e é a vez do próximo jogador jogar o mini jogo;

AwaitsExit

O estado *"AwaitsExit"* é responsável pelo final do jogo.

O utilizador pode efetuar as seguintes escolhas:

- **Sair da aplicação**;
- **Retornar ao Menu Inicial** - Ir para o estado *"AwaitsStart"* através da função *"goBack()"*;

AwaitsReplayID

O estado *"AwaitsReplayID"* é responsável pela escolha do replay que se pretende ver.

O utilizador pode efetuar as seguintes escolhas:

- **Voltar ao menu Inicial** - Ir para o estado *"AwaitsStart"* através da função *"goBack()"*;
- **Escolher o replay que se pretende jogar** - Ir para o estado *"AwaitsNextReplayMove"* através da função *"loadReplay(GameData gameData)"*;

AwaitsNextReplayMove

O estado “*AwaitsNextReplayMove*” é responsável pela demonstração do replay.

O utilizador pode efetuar as seguintes escolhas:

- **Voltar ao menu de escolha de Replay** - Ir para o estado “*AwaitsReplayID*” através da função “*goBack()*”;
- **Avançar para a próxima jogada do Replay** - Continuar no mesmo estado usando a função “*nextReplayMove(boolean isForward)*” sendo que a variável “*isForward*” é obrigatoriamente “*true*”;
- **Recuar para a jogada anterior do Replay** - Continuar no mesmo estado usando a função “*nextReplayMove(boolean isForward)*” sendo que a variável “*isForward*” é obrigatoriamente “*false*”;

Descrição das classes

FourInARowDemolisherEdition - classe que possui a função main para a interface de texto;

FourInARowDemolisherEditionGUI - classe que possui a função main para a interface gráfica;

Game - classe que serve de ligação entre o TextUI e a lógica interna do programa;

TextUI - classe que serve de interface de texto;

GameData - classe que possui métodos lógicos para o funcionamento do jogo.

GameBoardsFactory - classe que fabrica objetos do tipo GameBoard;

GameBoard - classe correspondente ao tabuleiro de jogo que implementa a interface IMementoOriginator;

GamePiecesFactory - classe que fabrica objetos do tipo GamePiece;

GamePiece - classe correspondente a uma peça de jogo;

NormalGamePiece - classe correspondente a uma peça de jogo normal que estende a classe GamePiece;

SpecialGamePiece - classe correspondente a uma peça de jogo especial que estende a classe GamePiece;

MiniGamesFactory - classe que fabrica objetos do tipo MiniGame;

MiniGame - classe abstrata correspondente a um mini jogo;

MathMiniGame - classe correspondente ao mini jogo de matemática que estende a classe MiniGame;

WritingMiniGame - classe correspondente ao mini jogo de escrita que estende a classe MiniGame;

PlayersFactory - classe que fabrica objetos do tipo Players;

Player - classe correspondente a um jogador;

HumanPlayer - classe correspondente a um jogador humano que estende a classe Player;

CPUPlayer - classe correspondente a um jogador virtual que estende a classe Player;

Memento - classe correspondente ao memento;

CareTaker - classe que implementa métodos que influenciam o jogo devido ao memento;

FileUtility - classe que trata Ficheiros;

StateAdapter - classe que implementa o interface das funções de estados;

AwaitsColumnChoice - classe correspondente a um estado;

AwaitsExit - classe correspondente a um estado;

AwaitsGameMode - classe correspondente a um estado;

AwaitsMathMiniGame - classe correspondente a um estado;

AwaitsMiniGame - classe correspondente a um estado;

AwaitsNextReplayMove - classe correspondente a um estado;

AwaitsPlayerName - classe correspondente a um estado;

AwaitsReplayID - classe correspondente a um estado;

AwaitsStart - classe correspondente a um estado;

AwaitsTheNextMove - classe correspondente a um estado;

AwaitsWritingMiniGame - classe correspondente a um estado;

InvalidColumnChoiceException - classe correspondente a uma exceção;

InvalidNumberOfCurrentPlayerIDException - classe correspondente a uma exceção;

InvalidNumberOfHumanPlayersException - classe correspondente a uma exceção;

InvalidTextUIOptionException - classe correspondente a uma exceção;

ImageLoader - classe responsável pelo carregamento de imagens;

Resources - classe com função para “ir buscar” um recurso pelo nome;

MainPane - classe com a organização mais geral da interface gráfica;

GUIConstants - classe com constantes que são usadas nos ficheiros de gestão da interface gráfica;

GameBoardPane - classe responsável pela apresentação do tabuleiro de jogo;

InfoPane - classe responsável pela apresentação de informações de jogo ao nível do jogador;

FIARDERoot - classe responsável pelo menu da barra superior da janela de interface gráfica;

AwaitsColumnChoicePane - classe responsável pela interface gráfica quando a máquina de estados se encontra no estado AwaitsColumnChoice;

AwaitsExitPane - classe responsável pela interface gráfica quando a máquina de estados se encontra no estado AwaitsExit;

AwaitsGameModePane - classe responsável pela interface gráfica quando a máquina de estados se encontra no estado AwaitsGameMode;

AwaitsMathMiniGamePane - classe responsável pela interface gráfica quando a máquina de estados se encontra no estado AwaitsMathMiniGame;

AwaitsMiniGamePane - classe responsável pela interface gráfica quando a máquina de estados se encontra no estado AwaitsMiniGame;

AwaitsNextReplayMovePane - classe responsável pela interface gráfica quando a máquina de estados se encontra no estado AwaitsNextReplayMove;

AwaitsPlayerNamePane - classe responsável pela interface gráfica quando a máquina de estados se encontra no estado AwaitsPlayerName;

AwaitsReplayIDPane - classe responsável pela interface gráfica quando a máquina de estados se encontra no estado AwaitsReplayID;

AwaitsStartPane - classe responsável pela interface gráfica quando a máquina de estados se encontra no estado AwaitsStart;

AwaitsTheNextMovePane - classe responsável pela interface gráfica quando a máquina de estados se encontra no estado AwaitsTheNextMove;

AwaitsWritingMiniGamePane - classe responsável pela interface gráfica quando a máquina de estados se encontra no estado AwaitsWritingMiniGame;

GameObservable - classe responsável pela ligação entre a lógica de jogo e a interface gráfica;

Constants - classe com uma constante para ser usada como property name;

Descrição do relacionamento entre classes

- A classe Game serve de ligação entre a classe TextUI e a classe GameData para que não haja troca direta de informação;
- As classes NormalGamePiece e SpecialGamePiece estendem a classe GamePiece;
- As classes MathMiniGame e WritingMiniGame estendem a classe MiniGame;
- As classes HumanPlayer e CPUPlayer estendem a classe Player;

Decisões tomadas na implementação

Foi dada a possibilidade de os jogadores humanos e virtuais poderem escolher um nome predefinido para além do nome personalizado na interface de texto.

Funcionalidades Implementadas

Todas as funcionalidades propostas no enunciado do trabalho prático para a 1º meta de entrega foram implementadas com sucesso.

Relativamente à 2º meta o jogo funciona na sua totalidade mas não é apresentada qualquer informação relativa ao tabuleiro e aos jogadores e são apresentados todos os botões apesar de alguns retornarem o mesmo estado quando são clicados porque não são possíveis de utilizar (Exemplo: CPU usar uma peça especial). É possível verificar que o jogo funciona ao fazer certas experiências tais como esperar 8 jogadas para aparecer um minijogo ou jogar com um jogador sempre na mesma coluna de modo a que faça os 4 em linha em jogador vs jogador.

Em suma, o jogo encontra-se implementado na totalidade com a ausência dos seguintes casos:

- apresentação das informações relativas ao tabuleiro e ao jogador atual na interface gráfica;
- não discriminação de botões inúteis para a jogada que retornam o estado atual (Exemplo: CPU usar peça especial);
- problema no acesso a um replay quando o botão correspondente é clicado na interface gráfica;

Conclusão

Este trabalho cultivou conhecimento relativo à matéria em questão fruto da prática da resolução do problema proposto.

A maior dificuldade concentrou-se na realização da interface gráfica.

O autor do trabalho conclui que o projeto foi realizado e resolvido com parcial sucesso devido a lacunas na interface gráfica.