

**Instituto Superior de Engenharia de Coimbra**

**Licenciatura em Engenharia Informática**

**Programação Avançada**

**Leandro Adão Fidalgo | a2017017144**

**Laboratório P3**

**Trabalho Prático 1**

**Meta 1**

**Coimbra, 23 de maio de 2021**

**Índice**

[1. Introdução 1](#_Toc29228)

[2. Conclusão 2](#_Toc9706)

1. Introdução

O presente relatório descreve o projeto desenvolvido pelo aluno: Leandro Fidalgo, no âmbito da disciplina de Programação Avançada da Licenciatura em Engenharia Informática do Instituto Superior de Engenharia de Coimbra.

A primeira meta do trabalho prático consistia na implenteação de toda a lógica do jogo em modo texto, suporte para dois jogadores humanos, humano vs. computador ou computador vs. Computador e ainda gravação ou carregamento do jogo (um jogo carregado de ficheiro deve permitir a sua continuação).

O objetivo deste trabalho consiste no jogo do 4 em linha com 2 mini jogos associados, um mini jogo dos cálculos e outro das palavras.

O objetivo do presente trabalho é consolidar todos os conhecimentos adquiridos nas aulas teóricas e práticas ao longo de todo o semestre.

1. Decisões tomadas na implementação

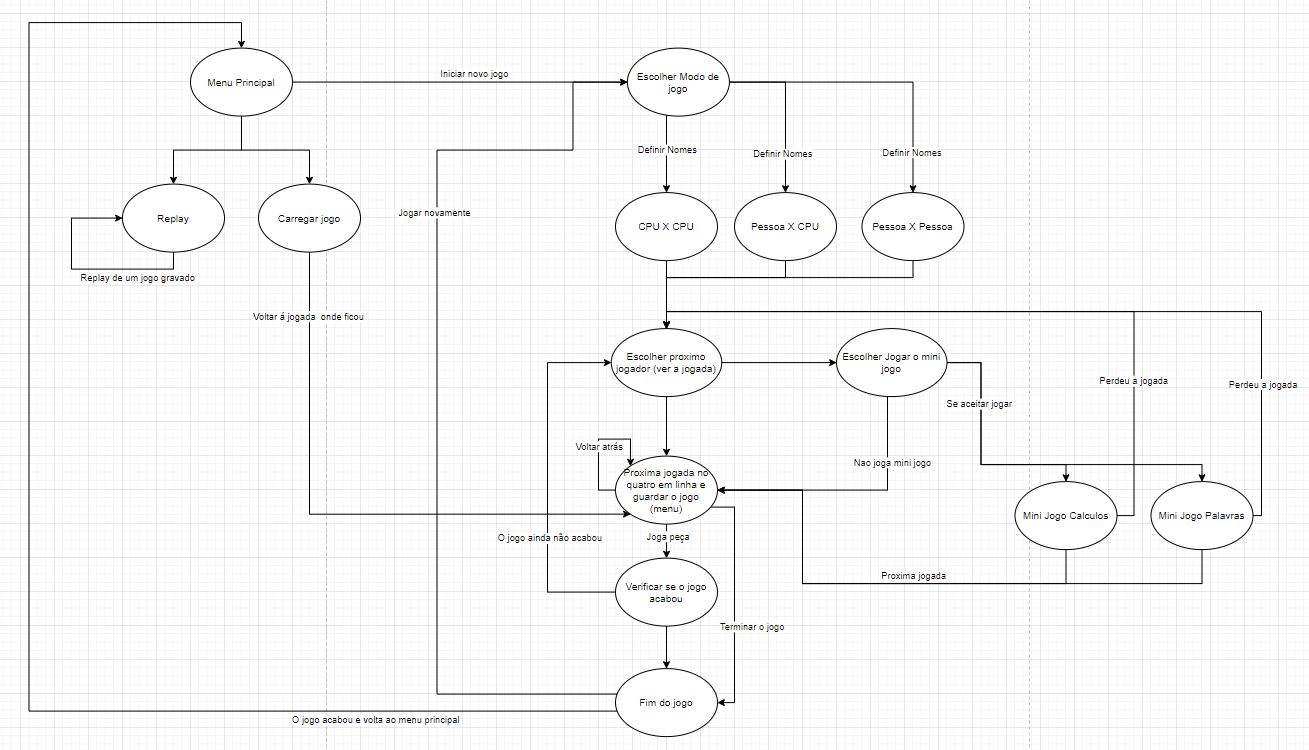
Durante a implementação do jogo foram surgindo algumas tomadas de decisão nomeadamente na forma de como é guardado o tabuleiro, foi optado por ser guardado num array bidimensional de inteiros, os Jogadores foram separados em Pessoa e CPU e foi ainda criada uma classe para os Minijogos para guardar informações relevantes aos mesmos.

Para voltar á jogada anterior durante o jogo foi utilizado uma Stack onde são guardados os tabuleiros ao longo do jogo e sempre que o jogador decide voltar á jogada anterior , dependendo do número de vezes que ele decide recuar é feito um pop á Stack.

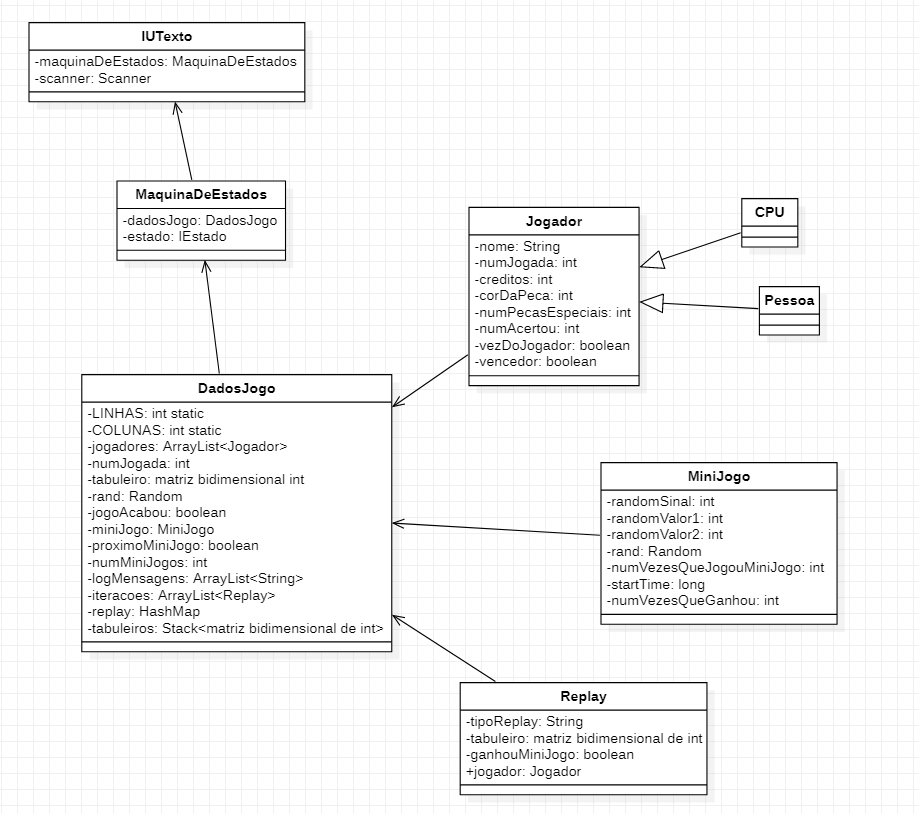
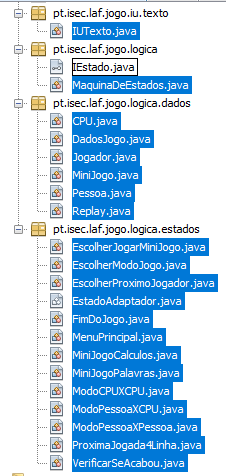
Para os Replays foi implementada uma classe onde são guardadas pequenas informações que irão ser guardadas para depois mostrar quando o Replay for efetuado. Os Replays são guardados num HashMap onde a key é sempre a data atual quando foi guardado o Replay e o value é o Replay, sendo este HashMap serealizado para um ficheiro chamado “Jogos” onde estarão sempre as informações relativas ao Replay.

Para guardar e carregar o jogo também são usados ficheiros, onde os DadosJogo são serealizados para dentro do mesmo.

1. Diagrama da máquina de estados



1. Descrição das classes



* 1. Classes relacionadas com a interface

Apenas existe uma classe relacionada com a interface sendo esta a Classe IUTexto e que tem toda a lógica relacionada com o que irá ser apresentado ao utilizador e relacionada com as introduções que o utilizador realiza. Esta classe contém a MaquinaDeEstados.

* 1. Classes relacionadas com a lógica

A única classe relacionada com a lógica do jogo é a MaquinaDeEstados e ainda uma interface para auxiliar nos diversos estados com o nome IEstado. A MaquinaDeEstados tem toda a lógica referente á mudança dos estados e ainda incorpora os DadosJogo.

* 1. Classes relacionadas com os dados

Nas classes relacionadas com os dados existem diversas. Porém a classe principal e que contém todas as informações é a classe DadosJogo, que contém um ArrayList de Jogador, que contém o MiniJogo e ainda o Replay. O CPU e a Pessoa são classes que fazem extend da classe Pessoa. Na classe DadosJogo é guardada toda a informação acerca do Jogo e esta classe é Serializable e mais tarde servirá para guardar os jogos.

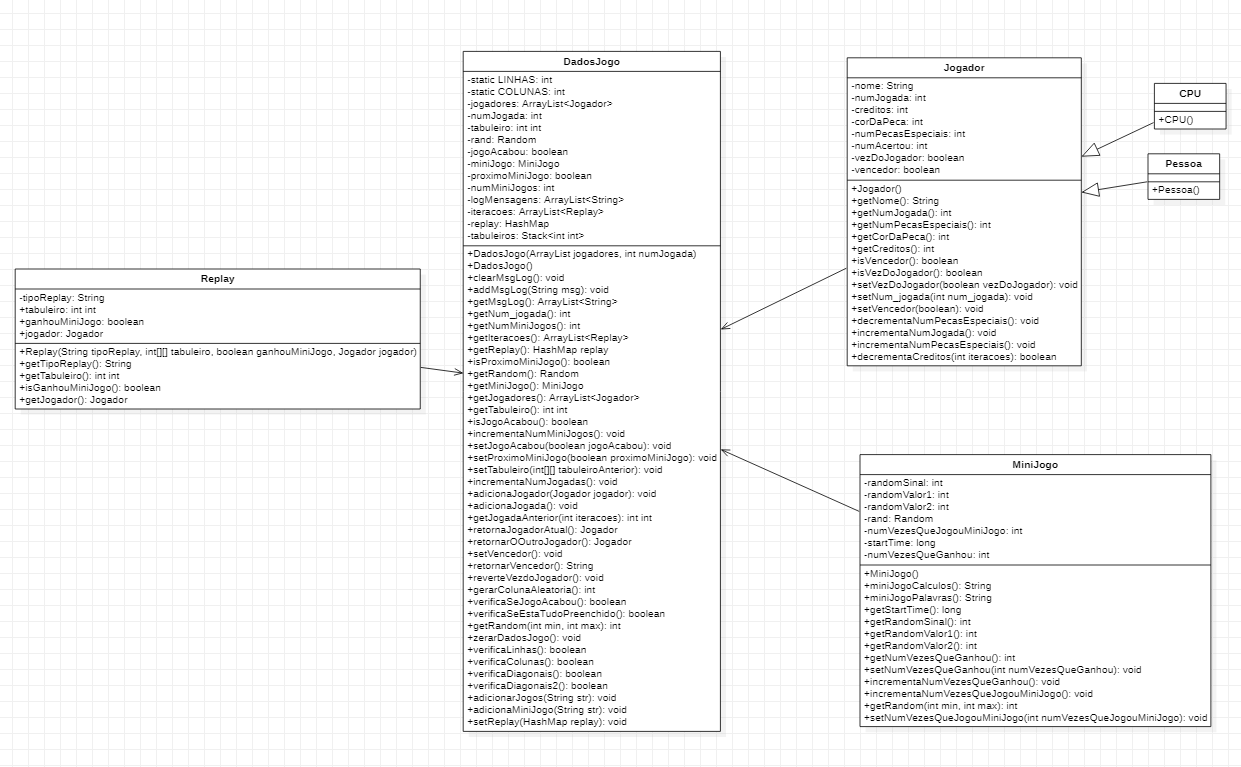
A classe MiniJogo serve para guardar informações acerca dos mini jogos, a classe Jogador serve para guardar os vários atributos dos jogadores e a classe Replay serve para guardar pequenas informações que serviram para fazer o replay de um jogo que estará guardado dentro de um ficheiro.

A classe DadosJogo ainda guarda os vários tabuleiros para quando é efetuada a operação de voltar á jogada anterior, contém o logMensagens que é onde são colocados os diversos logs e ainda contém um HashMap<String, ArrayList<Replay>> que irá ser usado para guardar os replays, sendo a String utilizada para guardar o nome do Replay e o ArrayList irá conter as várias informações que irão ser impressas ao utilizador quando fizer o replay.

* 1. Classes relacionadas com os estados

Nas classes relacionadas com os estados, a classe EstadoAdaptador serve para fazer a implementação das diversas funcionalidades que irão ser implementadas nas classes dentro do mesmo package, sendo estas classes os estados que foram apresentados na máquina de estados.

1. Descrição do relacionamento entre as classes



1. Funcionalidades

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. Conclusão

Com o desenvolvimento desta meta foi possível aprender muito sobre o conceito de máquina de estados entre outros conceitos abordados durante as aulas.

Durante o desenvolvimento desta meta foram surgindo problemas e desafios que foram superados com a ajuda dos professores da disciplina, os apontamentos por eles disponibilizados e da Internet.