***Jogo das Damas-Etapa1***

Carlos Nunes al67504

Diogo Fidalgo al65631

Luís Moreira al62679

Universidade de Trás-Os-Montes e Alto Douro

Licenciatura em Engenharia Informática

Docentes: André Sousa e Luís Barbosa

*Análise de Dados*

Depois da análise das regras do jogo das damas e do protocolo decidimos criar a implementação de treze classes começando por desenvolver as classes base Jogador, Jogo, Tabuleiro, Coordenada e Posição. Estas classes permitiram estruturar todo o nosso problema.

* *Menu* : A classe menu seria a primeira classe da consola e tem o objetivo de escolher uma opção de inicio de jogo-“standalone” onde se jogaria com um jogador virtual ou “cliente” á qual se jogaria com um jogador em rede. Para o modo de jogo pretendido a classe teria como atributo uma variável inteiro opcao.
* *Coordenada:* Esta classe possui dois atributos (x e y) do tipo inteiro que vão representar os valores da posição da matriz do tabuleiro.
* *Posição:* Esta classe representa a partir de duas heranças se uma coordenada (posição do tabuleiro) está ocupada com uma peça (classe vazia), se a casa do tabuleiro é de cor branca e por isso invalida para ser selecionada (classe invalida) . Possui o atributo coordenada do tipo coordenada para obter a localização da peça no tabuleiro e o atributo corfundo que irá armazenar a informação se a uma determinada casa do tabuleiro é de cor preta ( valor 1) ou branca (valor 0).
* *Peca:* A classe peça será herança a classe posição e servira para classificar as peças em relação á sua cor( preta ou branca) a partir do atributo bool corpeca , e em relação ao estado (comida, simples ou dama) .

È criado uma lista para armazenar o histórico das coordenadas do tabuleiro (list<Coordenada> histórico) para que possibilite saber se uma determinada peça numa certa posição existe, já foi comida ou se é uma dama e uma variável que guarda as jogadas sucessivas visto que o jogo acaba empatado após 20 jogadas sucessivas das damas sem tomada ou deslocamento de pedra.

Optámos que a classe “Peca” fosse uma subclasse da “Posicao”, para que assim recebesse por herança as coordenadas (o que vai ser útil noutras classes como a Jogador (“List<Peca> pecasEmJogo”)). Caso tivéssemos criado uma classe peça sem herança, teríamos que colocar como atributos as coordenadas e associar a uma subclasse da posição (que também já passa por herança essa informação), o que não seria muito viável.

De forma a distinguir as possíveis peças, decidimos criar (através de herança à classe “Peca”) as classes “Dama” e “Simples”. Usamos estas classes não só como forma de identificação no tabuleiro, mas também devido ao facto dos possíveis métodos movimento serem diferentes entre esses dois tipos de peças, uma vez que a peça dama tem mais alcance.

* *Tabuleiro:* A classe tabuleiro é representada por um conjunto de 64 posições (Do tipo “Posicao”). De forma a termos a informação necessária toda numa só matriz, decidimos que cada posição poderia ser de um de três tipos:
* ***Invalida****:* Posição do tabuleiro por onde as pecas não podem andar;
* ***Vazia****:* As peças podem ocupar essa posição, no entanto no momento atual não estão a ocupar a posição do tabuleiro;
* ***Ocupada*** (representada pela classe “Peca”): Em vez de criar-mos outra classe chamada Ocupada, optámos por criar a classe peça como herança da classe posição, para que desta forma, ao instanciar e inicializar o tabuleiro, apenas com uma matriz se tenha toda a informação necessários (Ex: coordenadas, se é valida ou vazia. Caso tenha peça, se é dama ou simples e quem a controla). Haveria outras formas de resolver parcialmente esta situação, como por exemplo com a utilização de *enums*, no entanto, para além de não resolver totalmente o problema, achámos esta a solução mais intuitiva.
* Jogo: Será responsável pelo desenrolar do jogo, ou seja, será responsável por saber quais são os jogadores que estão a jogar (jogador jogado1,jogador2) e pela gestão do tabuleiro no desenrolar do jogo. A classe “online” e “normal” herdam os atributos da classe jogo o que possibilita a repartição dos tipos de jogos com um acrecento dos atributos “string nomeServidor” e “bool autenticação” na classe “online já que é a inscrição dos jogadores e saber o nome do servidor é obrigatório.
* *Jogador:* Por fim, esta classe vai identificar qual é o tipo de peça(preta ou branca) que um jogador joga e o numero de peças em jogo e quais já foram eliminada. Caso um jogador pretenda recomeçar um jogo abandonado foi criado uma variável do tipo lista (List<Peca> pecasEmJogo) que representa as coordenadas do tabuleiro onde o jogador tem as suas pecas.

Uma vez que também era necessária a criação de um *bot*, decidimos criar uma classe “Robot” ligada por herança à classe “Jogador”. Com isto optamos por criar uma classe “pai” chamada *User,* da qual vão ser herdos os jogadores físicos e não os *bots,* através da introdução dos seus dados pessoais.