Mestrado Integrado em Engenharia Informática e Computação

2009/2010

Programação

Projecto nº 3 – Scrabble

Joaquim José Silva Faria Oliveira

José Carlos Portela Pereira

Turma 9

Grupo 3

29 de Maio de 2010

Introdução

No âmbito da unidade curricular de Programação foi-nos proposto no inicio da disciplina três projectos, sendo este o terceiro e último. Este projecto desafia-nos a desenvolver um jogo bastante conhecido, o "Scrabble", apesar de no primeiro projecto o termos desenvolvido, foi de uma maneira muito mais simples do que este terceiro projecto exige.

Enquanto que o "Scrabble" do projecto 1 consistiu no desenvolvimento de funções simples, em que apenas havia um jogador, com uma mão, pontuação, etc., desta vez, pretende-se que se desenvolva um programa muito mais complexo. Que vai desde a implementação de várias classes até á capacidade do nosso código conseguir guardar vários jogos, por exemplo.

Explicando, muito resumidamente, o jogo consiste no seguinte: temos dois jogadores (obrigatoriamente), e cada um vai receber um conjunto de 7 peças retiradas aleatoriamente de um saco que contêm várias peças. Com essas peças ele vai ter que formar palavras válidas, ou seja, que estejam no dicionário e que possam ser colocadas no tabuleiro, e consoante as peças que constituem a palavra vai receber uma determinada pontuação. O jogador para além de tentar formar palavras com as suas peças, pode passar a vez, ou também, trocar as letras que tenha na sua mão com as do saco, desde que tenha sempre 7 na sua mão. As palavras válidas são colocadas num tabuleiro 15 por 15, que contêm vários tipos de células, umas vazias, outras ocupadas e outras que contêm símbolos que funcionam como multiplicadores da pontuação. O jogo pode terminar de várias maneiras, por exemplo, quando um jogador passa a vez duas vezes consecutivas, quando algum jogador escolha sair do jogo, etc. No final do jogo, os jogadores são colocados num ranking, mediante as suas pontuações.

Neste projecto, há uma parte de valorização, que nos desafia a implementar uma funcionalidade adicional que diz respeito á inteligência artificial. Infelizmente, o nosso grupo não teve disponibilidade para fazer esta parte adicional.

Quanto á implementação de funcionalidades adicionais, o nosso grupo apenas implementou a básica.

Concepção e Implementação da Solução

Estrutura de Classes

A nossa estrutura de classes é constituída por 6 classes: “Peca”, “Jogador”, “Celula”, “Tabuleiro”, “Pontuacao” e “Jogo”.

A classe “Peca”, diz respeito ao conjunto de estruturas que formam a mão do jogador, cada uma dessas peças tem uma letra e elas estão colocadas num baralho onde irão ser retiradas aleatoriamente. Esta classe, tem funções, que obtêm a pontuação de uma peça ou o símbolo, por exemplo.

A classe “Jogador”, tem como funcionalidade fazer um jogador, com o seu respectivo, nome, mão e pontuação.

A classe “Celula”, tem como principal função constatar o estado do tabuleiro, se a célula for do tipo “char” quer dizer que a célula está vazia, ou seja, não tem lá nenhuma peça colocada, têm apenas um símbolo (multiplicador). Mas se tiver lá um apontador quer dizer que essa posição do tabuleiro está ocupada com uma peça do baralho.

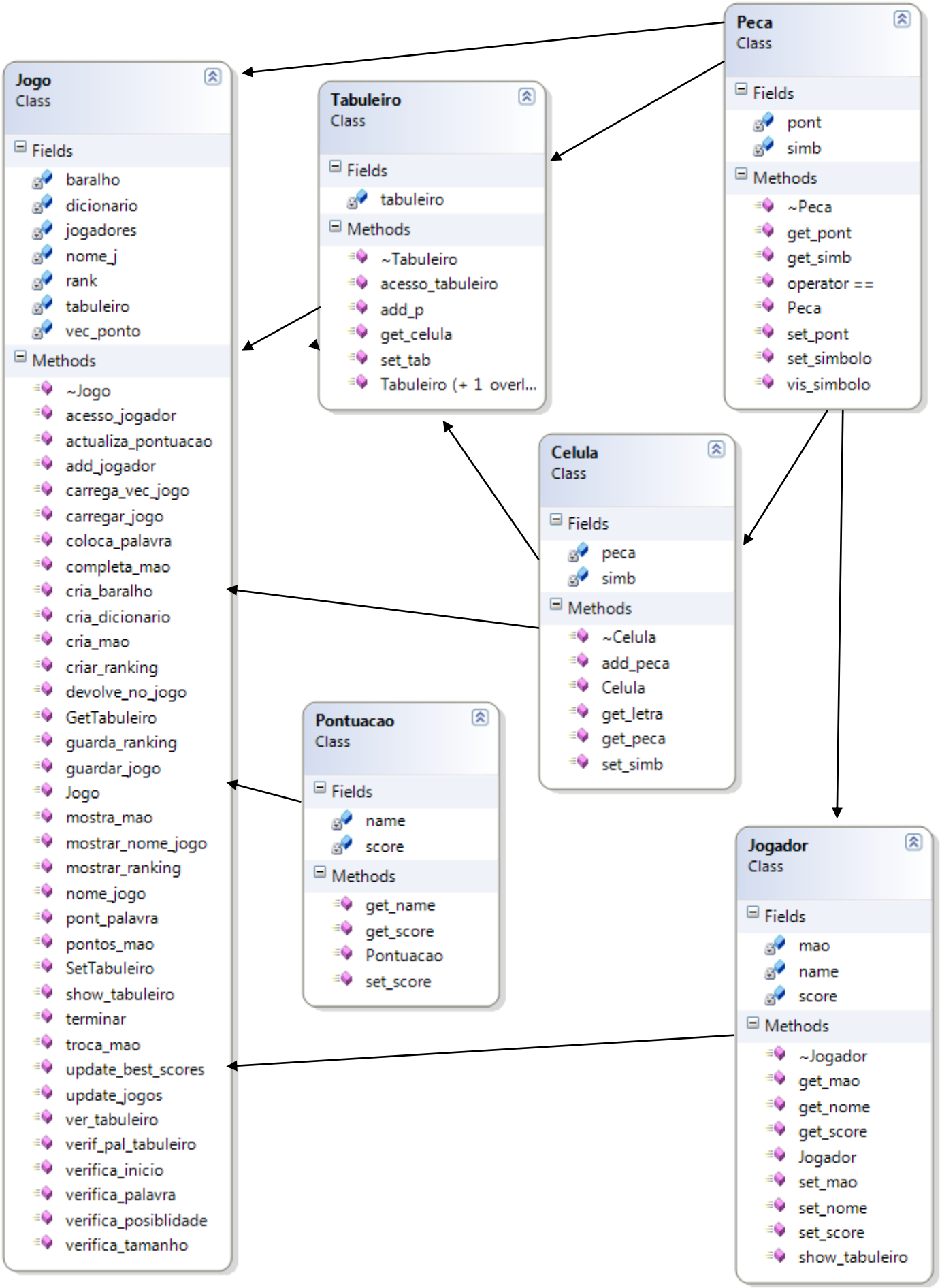
A classe “Tabuleiro”, que como nome indica, diz respeito ao tabuleiro 15 por 15 do Scrabble, tem várias funcionalidades como: adicionar uma peça a qualquer posição do tabuleiro, mostrar a partir de um ficheiro de texto a estrutura do tabuleiro, e também obter peças que estejam lá colocadas.

A classe “Pontuacao”, corresponde á pontuação dos jogadores. É uma classe simples, que tem como função obter e actualizar a pontuação dos jogadores ao longo do jogo.

A classe “Jogo”, que apesar de estar dependente das outras classes, é a nossa principal, porque contém as funções que vão ser essenciais para a função “main”. Nós optamos por, em vez de na função principal estar a fazer várias funções auxiliares á função “main”, definirmos quase todas as funções nesta classe. Esta, tem importantes funções, como por exemplo: cria o dicionário, essencial ao jogo, o tabuleiro, a mão do jogador, o baralho, também tem funções que se relacionam com o ranking dos jogadores, etc.

No diagrama de classes da página seguinte podemos observar relações entre as várias classes: a “Jogo”, como é a classe que contêm as principais funções que vão ser usadas na função “main”, necessita de recorrer a todas as outras; a “Peca”, constitui a nossa classe-base (foi a partir desta que começamos a construir o nosso programa), por isso, não precisa de fazer nenhum “include” de outra classe; a “Jogador”, recorre ás classes “Peca” e “Pontuacão”, porque a informação que precisamos para adicionar um novo jogador é apenas: o seu nome, a sua mão (que é um vector de peças) e a sua pontuação; tal como a “Peca”, a classe “Pontuacao” não recorre a nenhuma outra classe; a “Celula”, apenas recorre á classe “Peca”, porque no jogo em si vão estar relacionadas, as peças são colocadas nas células do tabuleiro; finalmente a classe “Tabuleiro”, recorre á “Peca” e á “Celula”, porque, logicamente, o tabuleiro é um conjunto de células e as peças são colocadas nestas.

Diagrama de Classes



Implementação das classes e algoritmos utilizados

No processo de desenvolvimento da nossa estrutura de classes, esta, ao longo do desenvolvimento do nosso projecto, foi sofrendo várias alterações, isso verifica-se ao compararmos as classes que estão no nosso diagrama de classes da parte intermédia do projecto com as de agora. Um exemplo concreto é, na parte intermédia tínhamos mais uma classe, a "Mao", mas reflectindo melhor constatamos que a mão do jogador iria ser um vector de peças, então optamos por eliminar essa classe e manter, apenas, a classe "Peca".

Vamos fazer uma curta explicação das funções, que implementamos após a entrega da parte intermédia do projecto, ou seja, que não foram colocadas no diagrama de classes:

- **void Jogo::add_jogador**(Jogador player): esta função coloca um jogador no vector dos jogadores;

- **bool Jogo::verifica_posiblidade**(string palavra, **int** pos_horiz, **int** pos_vert, **char** direc): função que verifica se na posição que a palavra vai ser colocada existe alguma peça (apontadores) lá, de modo, a que a palavra possa ser colocada no tabuleiro;

- **bool Jogo::verif_pal_tabuleiro**(string palavra, **int** pos_horiz, **int** pos_vert, **char** direc, **int** i) : esta função verifica se uma palavra pode ser colocada no tabuleiro, com as peças já existentes no tabuleiro e com as que o jogador tem na mão;

- **void Jogo::coloca_palavra**(string palavra, **int** pos_vert, **int** pos_horiz, **char** direc, vector<Peca> & mao): coloca uma palavra no tabuleiro;

- **void Jogo::update_best_scores**(Pontuacao s): função que actualiza o ranking do jogo com as novas pontuações dos jogadores;

- **int Jogo::pontos_mao**(**int** i): esta função conta os pontos da mão do jogador após o final do jogo.

A parte do projecto em que talvez tenhamos sentido mais dificuldade foi na de guardar/carregar os vários jogos, porque não sabíamos o modo como iríamos proceder para guardar tanta informação. Resolvemos este problema agrupando os dados em grupos.

O menu inicial indica três possibilidades. No primeiro caso inicia um novo jogo, no segundo carrega um jogo guardado anteriormente em um ficheiro de texto, e o terceiro sai do jogo.

Todos os casos onde é pedido a intervenção do jogador estão preparados caso o pedido não for cumprido, impedindo assim que o programa de erro.

Quando colocamos uma peça no tabuleiro esta e testada antes em relação à sua existência, tamanho, posição no dicionário e se é possível formar com as letras que temos na mão e as já existentes no tabuleiro.

Conclusão

O que mais nos intrigou ao concluirmos o nosso projecto foi não termos conseguido (não por falta de vontade) implementar a parte da inteligência artificial. A principal causa foi por termos gerido mal o tempo, principalmente, devido a outras tarefas propostas por outras unidades curriculares e também devido á dificuldade que essa parte adicional exigia.

Conseguimos cumprir todas as funcionalidades que o enunciado nos pedia, o programa "corre" perfeitamente e sem qualquer limitação.

Achamos que o programa poderia ser melhorado, se tivesse interface gráfica. Mas isso ainda não aprendemos, logo, temos que ser pacientes e aguardar por tempos futuros…