Projeto da Disciplina - Competição de quatro em uma linha (*Four-in-a-row*) - Versão 2008/2

Prof. Fabrício Jailson Barth
Laboratório de Programação IV - IA
Centro Universitário SENAC

2^o. semestre de 2008

1 Definição do jogo

O jogo quatro em uma linha[1] (*Connect Four*) é um jogo de tabuleiro de duas pessoas. As jogadas são alternadas entre os jogadores. Em cada jogada cada jogador escolhe uma coluna para inserir uma bola com a cor do jogador. As bolas são inseridas em um tabuleiro vertical com sete colunas e seis linhas. O objetivo do jogo é ligar quatro bolas da mesma cor em uma linha vertical, horizontal ou diagonal - antes que o seu oponente faça isso[2]. Um exemplo de fim de jogo é apresentado na figura 1.

O objetivo deste trabalho é implementar um jogador para este jogo. Este jogador deverá participar de uma competição. As regras e restrições da competição são apresentadas na próxima seção.

2 Regras e restrições da competição

A competição é do tipo *todos-contra-todos* com jogos de ida e volta. Ou seja, todo jogador irá jogar contra todos os outros jogadores duas vezes. Os resultados podem ser vitória, derrota ou empate. A vitória vale 1 ponto, empate vale 0.5 e derrota vale 0. Os pontos de cada jogador serão somados depois do término dos jogos.

A competição terá como participantes os jogadores implementados pelas equipes, um jogador *aleatório*, um jogador *avançado* desenvolvido pelo professor e outros três jogadores vencedores do semestre passado. Informações sobre como os três jogadores vencedores do semestre passado foram implementados poderão ser encontradas nos relatórios anexos.

Nenhum jogador poderá perder do jogador chamado *"aleatório"* (*baseline* da competição). A equipe que perder do *baseline* da competição terá a sua pontuação na competição decrementada por 3 pontos.

Cada jogador deve realizar as jogadas em menos de 10 segundos. Se o jogador realizar uma jogada que demore mais que 10 segundos então a equipe terá a sua pontuação na competição decrementada por 2 pontos.

[1] Também conhecido como *Conecta Quatro*
[2] http://en.wikipedia.org/wiki/Connect_Four

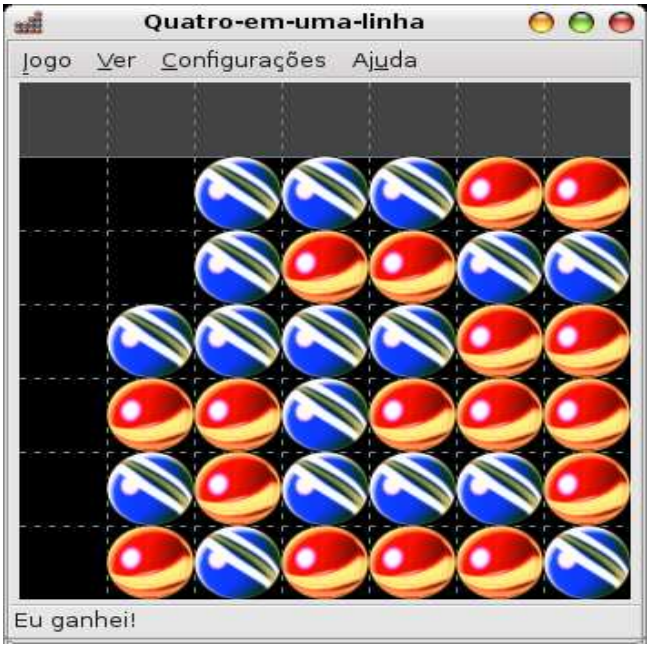

Figura 1: Exemplo de fim de jogo

O jogador que provocar o *looping* infinito do gerenciador terá a sua pontuação na competição decrementada por 5 pontos. Este tipo de situação pode acontecer se o jogador for programado para escolher sempre a mesma coluna ou sempre as mesmas poucas colunas.

Após o término da competição, os pontos de cada jogador serão somados. Quanto maior o número de pontos, melhor será a colocação do jogador e, conseqüentemente, melhor será a nota da equipe que desenvolveu o jogador. Cada equipe poderá ser formada por até três pessoas.

3 Utilização do ambiente para a competição

Será disponibilizado para as equipes um pacote com as seguintes classes:

- *GerenciadorLinhaQuatro*: Trata-se da classe responsável por gerenciar os jogos. Os jogos realizados estão *hard-coded*. Cada nova seqüencia de jogos (campeonato) deve ser codificada nesta classe.
- *Jogador*: Interface responsável por definir os métodos que devem ser implementados pelos jogadores.
- *JogadorAleatorio*: Implementação do jogador "aleatório", *baseline* da competição.
- *JogadorAleatorioFocado*: Implementação de um outro tipo de jogador aleatório que joga apenas nas casas 3, 4 e 5, nas casas centrais.
- *JogadorManual*: Implementação de um jogador que espera pela entrada do usuário.

- *JogoLinhaQuatro*: Classe que implementa a lógica do jogo.

Os arquivos que merecem atenção das equipes são *GerenciadorLinhaQuatro.java* e *Jogador.java*. Na próxima seção é apresentado um roteiro para implementação dos jogadores.

4 Implementação dos jogadores

Cada equipe deve criar uma classe que representa um jogador. Esta classe deve implementar a interface *Jogador*:

```
public class MeuJogador implements Jogador
```

A interface *Jogador* é composta pelos seguintes métodos:

- *public String getNome()*: este método deve retornar o nome do jogador. Utilize um nome significativo, pois este nome será utilizado para gerar o arquivo de *log* da competição.
- *public int jogada(int[][] tabuleiro, int corDaMinhaBola)*: este método recebe a situação atual do tabuleiro, a cor da bola do jogador (1 ou 2) e retorna a coluna onde a bola deve ser colocada $(0, 1, \cdots, 6)$.

Um exemplo de classe que implementa um jogador é apresentado no código 2.

```
1  public class JogadorAleatorio implements Jogador{
2
3          public int jogada(int[][] tabuleiro, int corDaMinhaBola) {
4                  double value = Math.random();
5                  if(value>=0.0 && value<=0.14)
6                          return 0;
7                  else if(value>=0.15 && value<=0.29)
8                          return 1;
9                  else if(value>=0.3 && value<=0.44)
10                         return 2;
11                 else if(value>=0.45 && value<=0.59)
12                         return 3;
13                 else if(value>=0.6 && value<=0.74)
14                         return 4;
15                 else if(value>=0.75 && value<=0.89)
16                         return 5;
17                 else
18                         return 6;
19         }
20
21         public String getNome() {
22                 return "Aletório";
23         }
24 }
```

Figura 2: Implementação da classe JogadorAleatorio

A classe que implementa o jogador da equipe pode fazer referência a outras classes necessárias para o raciocínio do jogador.

4.1 Sugestões para testes

Sugere-se para testes realizar algumas competições com outros jogadores disponibilizados no pacote. A cada nova versão de jogador, pode-se utilizar a versão antiga para uma competição. Sugere-se fortemente, no mínimo, testar o jogador contra o jogador "aleatório" (*baseline* da competição).

Para criar novas competições é necesário alterar a classe *GerenciadorLinhaQuatro* e executá-la.

5 Outros Jogadores (participantes do campeonato)

Os outros jogadores participantes da competição são apresentados abaixo:

- **TUX**: Jogador desenvolvido pelos alunos Diego Ucha e Fábio Montefusco no 2^o semestre de 2007.
- **BGM**: Jogador desenvolvido pelos alunos Bruno Herrera, Gabriel Koji e Marcelo Honório no 2^o semestre de 2007.
- **Timão**: Jogador desenvolvido pelos alunos Caio Sanchez e Alexandre Sierra no 1^o semestre de 2008.
- **"Avançado"**: Jogador desenvolvido pelo professor Fabrício.

Todos os relatórios sobre como os jogadores foram desenvolvidos podem ser encontrados na pasta *doc/descricaoJogadores*.

6 Avaliação e formato do relatório

A avaliação do trabalho é formada pelos seguintes itens:

- Colocação do jogador na competição ($0-7$ pontos). O jogador que obtiver a melhor pontuação receberá 7 pontos. A segunda melhor pontuação receberá 6 pontos. A cada nível de pontuação será decrementado 1 ponto da equipe. Um exemplo pode ser visualizado na tabela 1.

Tabela 1: Exemplo de pontuação

Jogador	Pontos obtidos na competição	Nota
$Jogador_1$	3	6
$Jogador_{baseline}$	0	sem nota
$Jogador_3$	6	7
$Jogador_4$	3	6

- Análise do relatório ($0-3$ pontos). Será avaliado a clareza, objetividade e completude de cada item do relatório. O relatório deve conter os seguintes itens:

- **Introdução e Método**: descrever quais foram os passos realizados para chegar até a solução.
- **Solução**: a descrição da solução adotada. Que algoritmo foi implementado e quais os detalhes de implementação. Por exemplo, se o algoritmo utilizado for o MIN-MAX, detalhe o máximo possível a **função de utilidade** e descreve qual foi a profundidade adotada.
- **Testes**: Descrever quais foram os testes realizados antes da competição para garantir que o seu jogador funcione. Descrever quais foram os testes realizados antes da competição para inferir a eficiência do jogador. Descrever os testes para determinar os melhores valores de parâmetros para o jogador.
- **Considerações finais**: Descrever os itens que fazem você acreditar que o seu jogador terá um desempenho bom, ou até mesmo ganhe a competição.

O relatório deverá ser entregue em um formato manipulável por *Copy-Paste*: doc, odt, LaTeX, por exemplo. O conteúdo do relatório será utilizado na confecção de um relatório único e final da competição.

Utilize o relatório do jogador "Avançado" como exemplo.

- Análise do projeto do jogador e do código fonte. Este item da avaliação tem como objetivo identificar cópias de trabalhos e incoerências entre a implementação e o relatório.

  Se for identificado cópia de trabalhos, a nota do projeto será calculada da seguinte maneira:

$$Nota\_Final = \frac{Nota\_Da\_Competicao + Nota\_Do\_Relatorio}{Numero\_Trabalhos\_Identicos} \quad (1)$$

  para todos os trabalhos identificados como similares.

  Se for identificado incoerências graves entre o relatório e a implementação, a nota da equipe será zero. Incoerências leves irão decrementar a nota da equipe.

  Em casos normais, a nota será calculada da seguinte maneira:

$$Nota\_Final = Nota\_Da\_Competicao + Nota\_Do\_Relatorio \quad (2)$$

7 Nota importante

Existem trabalhos, de mestrado e doutorado - disponíveis na Internet, que provam que existe uma forma para sempre ganhar neste jogo. Seria muito interessante se uma, ou mais, equipes conseguissem implementar um jogador que sempre ganha para este jogo.