Especificação do Trabalho 01

Descrição inicial

Modele o diagrama de classes em UML e implemente uma aplicação em Java para um jogo de dominó (2 a 4 participantes). A implementação deve oferecer dois modos de execução:

1. Um jogador humano contra um, dois ou três jogadores simulados pela aplicação.
2. Dois, três ou quatro jogadores simulados pela aplicação (modo demonstração).

Quando for o caso, deve ser perguntado ao jogador humano seu nome. Os jogadores simulados devem ser nomeados automaticamente (“Roy Batty”, “Leon Kowalski”, “Pris Stratton” e “Zhora Salome”).

Para este trabalho, adotaremos o conjunto tradicional de dominós (duplo-seis), conhecido como sino-europeu, formado por 28 peças (pedras). Cada face retangular de dominó é dividida em duas partes quadradas (pontas), que são marcadas por um número de pontos de 0 a 6, ou deixadas em branco.

Informações sobre o jogo podem ser encontradas em http://pt.wikipedia.org/wiki/Domin%C3%B3.

Orientações sobre o jogo

Fonte: http://www.jogos.antigos.nom.br/domino.asp

Cada jogador recebe 7 peças, que mantém escondida dos olhos do adversário.

Inicia o jogo quem tiver o duplo-6 (peça com o número 6 nas suas duas metades), também conhecida como "carroça" ou "carroção". A lista abaixo mostra a sequência a ser utilizada (em ordem crescente de peso). Ou seja, caso nenhum jogador tenha a peça [6,6], aquele que tiver a peça [5,5] deve iniciar. Caso contrário, aquele que tiver a peça [4,4] e assim por diante.

[0,1], [0,2], [0,3], [0,4], [0,5], [0,6]

[1,2], [1,3], [1,4], [1,5], [1,6]

[2,3], [2,4], [2,5], [2,6]

[3,4], [3,5], [3,6]

[4,5], [4,6]

[5,6]

[0,0], [1,1], [2,2], [3,3], [4,4], [5,5], [6,6]

A partir de quem iniciou, cada jogador, em ordem horária, colocará uma peça que se encaixe em uma das "pontas" da cadeia que vai se formando com as peças que vão sendo colocadas. Se alguém não tiver peça a colocar, vai ao "monte" e "compra" até conseguir uma peça que sirva. Caso não exista tal peça, o jogador "passa" sua vez ao jogador seguinte. Vence quem se livrar de todas as suas peças. No caso do jogo ficar "travado", isto é, não houver possibilidade de se colocar peças, contam-se os pontos nas mãos de cada jogador. Vence aquele que tiver menor número de pontos na mão.

Especificação do Trabalho 01

Orientações para o desenvolvimento

Ao final de cada jogada, o programa deve mostrar informações sobre o jogador e sobre a situação da mesa (ou tabuleiro), conforme o exemplo abaixo:

```
Rodada     : 1
Jogador    : Roy Batty
Mão        : [1,2] [0,2] [0,6] [1,3] [4,6] [2,5]
Compradas  :
Pedra usada: [6,6]
Mesa       : [6,6]

Rodada     : 2
Jogador    : Marcello
Mão        : [0,1] [4,5] [2,2] [0,5] [3,5] [1,5]
Compradas  :
Pedra usada: [3,6]
Mesa       : [3,6][6,6]
```

Dicas

- Considere boas práticas discutidas durante a disciplina e no material fornecido: coesão (classe e métodos), separação entre E/S, processamento/controle e entidades (veja o exemplo entregue no roteiro 02 para buscar inspiração). Considere também a organização dos pacotes (packages).
- Utilize javadoc para a documentação do código (leia a seção de comentários dos slides de referência disponibilizados no material didático).
- Não é necessário trabalhar com interface gráfica. Utilize a classe `Console` fornecida no roteiro 02. Se você tem dúvidas sobre esta classe, leia seu javadoc e revise os slides de referência para conhecer mais sobre a classe `Scanner`.
- Pense na representação de cada peça.
- Pense em como sortear as peças e distribui-las entre os jogadores.
- Pense em como saber, a cada jogada, se uma determinada peça pode ser encaixada no dominó. Não se esqueça que em um dominó você pode encaixar uma peça na frente ou atrás, mas não no meio.
- Para jogadores simulados pela aplicação, utilize uma lógica simples para selecionar a peça: escolher a primeira peça que se encaixe naquela jogada. Se não houver peças disponíveis, realizar a compra de peças até que uma se encaixe. Se não houver mais peças para comprar e nenhuma se encaixar, então o jogador deve "passar" (a vez é passada para o próximo jogador).
- No caso do jogador humano, você deve perguntar qual peça ele deseja utilizar (não esqueça de garantir que a peça seja válida).
- Para saber se o jogo acabou, verificar se todos os jogadores não conseguem mais encaixar nenhuma peça (todos passaram) ou então um dos jogadores ficou sem peças (bateu). Para o primeiro caso, o vitorioso é aquele que tiver o menor somatório de pontos (soma dos lados das peças restantes).
- Quando o modo de jogo escolhido for aquele com todos os jogadores simulados, as rodadas devem ser apresentadas até o final do jogo, sem a necessidade de interação com o usuário. Neste modo de jogo, o programa deve apresentar também as peças que estão no monte a cada rodada (incluir uma linha após a impressão das peças da mesa).
- Para aqueles que quiserem implementar o código em inglês, veja algumas referências em https://www.pagat.com/tile/wdom/all_fives.html,

Especificação do Trabalho 01

https://www.gamecolony.com/domino_game_rules.shtml e https://www.dominorules.com/straight-dominoes. Para facilitar, algumas dicas:

- Conjunto de dominós duplo-seis: *double six domino set*
- Pedra: *tile* ou *bone*
- Lado ou ponta (da pedra): *end*
- Mão: *hand*
- Jogador: *player*
- Resto ou Monte (somente quando o número total de jogadores for inferior a 4): *boneyard*
- Embaralhar: *shuffle*
- Comprar: *draw*
- Bateu: *dominoed*
- Passou (não tem pedras e não há mais pedras para comprar): *blocked*
- Mesa ou tabuleiro: *table*
- Pontuação: *scoring*
- Rodada ou vez: *turn*