Terceiro Exercício-Programa

Luciano Antonio Digiampietri
Norton Trevisan Roman

Prazo máximo para a entrega: 01/06/2020

1 Jogo da Velha

O Jogo da Velha, Jogo do Galo ou Jogo das Três Linhas é um jogo bastante popular com regras simples. Ele é jogado por dois jogadores, sendo composto por um tabuleiro com nove casas, arranjadas em três linhas e três colunas. O tabuleiro é iniciado vazio, e em cada jogada um jogador coloca uma "peça" em uma das casas vazias. Vamos considerar que o primeiro jogador possui a peça X (representado neste EP pela letra 'x' maiúscula) e o segundo jogador possui a peça O (representado neste EP pela letra 'o' maiúscula).

Consideraremos que o jogo sempre será iniciado pelo jogador com a peça X. O jogo acaba assim que um dos jogadores conseguir colocar três de suas peças (três X ou três O) em uma sequência: na mesma linha, na mesma coluna ou em uma das diagonais. Neste caso, o jogador que conseguiu produzir essa sequência será o ganhador. A outra alternativa para o fim do jogo é as nove casas do tabuleiro serem preenchidas com X e O sem nenhuma das sequências vitoriosas acontecerem. Neste caso, o jogo acabará em empate[1][2].

Neste EP você deverá implementar um método que verifique o *status* de um tabuleiro válido durante um Jogo da Velha. Os possíveis estados do tabuleiro são:

- 0 – Jogo não iniciado: o tabuleiro está "vazio", isto é sem peças X e O;
- 1 – Jogo encerrado 1: o primeiro jogador (que usa as peças X) é o ganhador;
- 2 – Jogo encerrado 2: o segundo jogador (que usa as peças O) é o ganhador;
- 3 – Jogo encerrado 3: empate – todas as casas do tabuleiro estão preenchidas com X e O, mas nenhum dos jogadores ganhou;
- 4 – Jogo já iniciado e em andamento: nenhuma das alternativas anteriores.

Você deverá implementar um método com a seguinte assinatura:

```
static int verificaStatus(char[][] tabuleiro)
```

[1] `https://pt.wikipedia.org/wiki/Jogo\_da\_velha`
[2] `https://en.wikipedia.org/wiki/Tic-tac-toe`

Este método receberá um tabuleiro no formato de uma matriz 3×3 de caracteres. Você pode assumir que todos os tabuleiros sempre corresponderão a jogos válidos (isto é, sempre estarão preenchidos com X, O ou espaço em branco (caractere `' '`, e não o caractere vazio `''`, para representar casas vazias/livres); os jogadores alternarão entre suas jogadas e não é possível em um único jogo haver dois ganhadores.

1.1 Entrada

A entrada é composta pelo parâmetro do método, que recebe uma matriz de caracteres com dimensões 3×3, correspondendo a um tabuleiro de jogo da velha.

1.2 Saída

O método deverá retornar um número inteiro (entre 0 (zero) e 4, correspondendo ao *status* do jogo do tabuleiro atual, conforme a lista de estados possíveis já apresentada).

Para que você teste o funcionamento do método `verificaStatus`, pode fazer o método `main()` escrever na tela o valor do *status* da execução do método para diferentes tabuleiros de entrada. No código fornecido juntamente com o enunciado já são apresentados alguns exemplos (se desejar imprimir coisas na tela, faça isso **apenas no método main** durante seus testes. **Não use `println()` diretamente em `verificaStatus`**).

1.3 Material a Ser Entregue

Um arquivo, denominado `JogoDaVelha.java`, contendo o método `verificaStatus()` e qualquer outro método que ache necessário. Para sua conveniência, `JogoDaVelha.java` será fornecido, cabendo a você então completá-lo.

Atenção!

1. Não modifique a assinatura de `verificaStatus()`!
2. Para avaliação, apenas o método `verificaStatus()` será invocado diretamente. Em especial, qualquer código dentro do `main()` será ignorado. Então certifique-se de que o problema seja resolvido chamando diretamente somente esse método.

2 Entrega

A entrega será feita única e exclusivamente via eDisciplinas, até a data final marcada. Deverá ser postado no eDisciplinas um arquivo zip, tendo como nome seu número USP:

`número_usp.zip`

Dentro do zip deve constar tão somente o arquivo `JogoDaVelha.java` com seu código nele. Não esqueça de preencher o cabeçalho constante do arquivo, com seu nome, número USP, etc.

A responsabilidade de postagem é exclusivamente sua. Por isso, submeta e certifique-se de que o arquivo submetido é o correto (fazendo seu download, por exemplo). Problemas referentes ao uso do sistema devem ser resolvidos com antecedência.

3 Avaliação

Para avaliação, serão observados os seguintes quesitos:

1. Documentação: se há comentários explicando o que se faz nos passos mais importantes e para que serve o programa (tanto o método quanto o programa em que está inserido);
2. Apresentação visual: se o código está legível, indentado, etc;
3. Corretude: se o programa funciona.

Além disso, algumas observações pertinentes ao trabalho, que influem em sua nota, são:

- Este exercício-programa deve ser elaborado individualmente;
- Não será tolerado plágio, em hipótese alguma;
- Exercícios com erro de sintaxe (ou seja, erros de compilação), receberão nota ZERO.