

Trabalho MC322

Etapa 03

Jogo de Xadrez

O seu desafio aqui é escrever um conjunto de classes que simule o funcionamento de um jogo de xadrez. Este jogo tomará como base o jogo damas e deve seguir duas premissas do mesmo:

- defina pelo menos uma classe cujo objeto represente o tabuleiro;
- defina uma classe que represente cada tipo de peça do tabuleiro.

Tal como nas damas, como o jogo de xadrez tem vários tipos de peça, deve ser feita uma hierarquia de classes com uma classe que represente uma peça genericamente (com as características comuns de qualquer peça) e uma subclasse que represente cada tipo de peça.

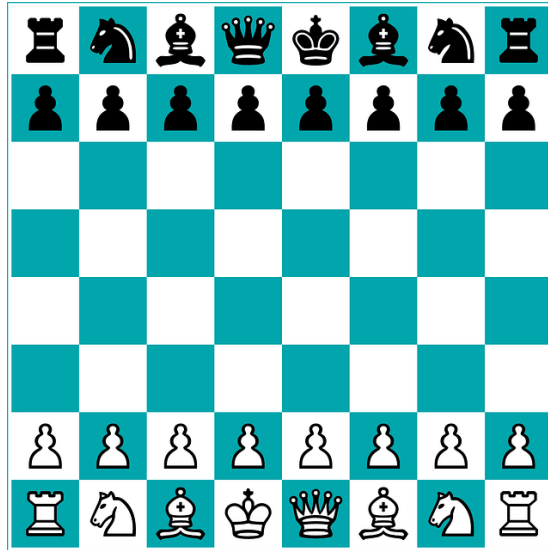
A requisição de movimento será feita obrigatoriamente à peça. A peça verificará se o movimento é compatível com o seu tipo e se o movimento é possível no tabuleiro. Para verificar se o movimento é possível, a peça se comunica com o tabuleiro, que pode lhe informar, por exemplo, o que há em certas posições do tabuleiro.

Se a peça verificar que é possível, a peça comunica ao tabuleiro o movimento desejado. O tabuleiro executa o movimento, retirando a peça da origem e colocando no destino.

Nesta versão do xadrez, deve ser explorado o polimorfismo, ou seja, a peça genérica deve definir métodos que sejam sobrescritos pelas subclasses, fazendo com que elas se comportem de forma polimórfica.

A verificação de captura deve ser feita pela peça, que comunica ao tabuleiro a captura, que por sua vez executa a remoção da peça.

Trata-se de um tabuleiro clássico de xadrez:



- Lance:
 - o jogo está organizado em rodadas (lances) em que os oponentes se revezam em movimentos no tabuleiro;
- Movimento:
 - peças se movem conforme as regras do xadrez.
- Transformação do peão: quando um peão atinge o lado oposto do tabuleiro ele se transforma em uma peça à escolha do jogador.
- Captura da oponente:
 - segue as regras do xadrez.

Os movimentos a serem jogados no tabuleiro serão recebidos a partir da chamada de um método de um objeto, cuja classe já está codificada (veja detalhamento abaixo).

Tabuleiro

O estado inicial do tabuleiro e seu novo estado depois de cada movimento deve ser mostrado no console na forma de caracteres, como ilustrado a seguir:

An 8x8 chessboard with alternating light blue and white squares. The 8th rank (top) contains black pieces: Tower at a8, Horse at b8, Bishop at c8, Queen at d8, King at e8, Bishop at f8, Horse at g8, and Tower at h8. The 1st rank (bottom) contains white pieces: Tower at a1, Horse at b1, Bishop at c1, King at d1, Queen at e1, Bishop at f1, Horse at g1, and Tower at h1. All other squares are empty.

8	T	H	B	K	Q	B	H	T
7	P	P	P	P	P	P	P	P
6	-	-	-	-	-	-	-	-
5	-	-	-	-	-	-	-	-
4	-	-	-	-	-	-	-	-
3	-	-	-	-	-	-	-	-
2	p	p	p	p	p	p	p	p
1	t	h	b	q	k	b	h	t
	a	b	c	d	e	f	g	h

T (tower) - torre
H (horse) - cavalo
B (bishop) - bispo
K (king) - rei
Q (queen) - rainha
P (paw) - peão
Utilize maiúsculas para representar as peças pretas e minúsculas para representar as brancas.

A ilustração também mostra o estado inicial do tabuleiro.

O tabuleiro deverá ser apresentado em caracteres da forma mostrada acima. As posições do tabuleiro serão descritas utilizando as colunas de **'a'** até **'h'** e as linhas de **1** até **8**, onde uma posição é dada combinando coluna com linha, por exemplo, **c4** consiste na coluna **c** e na linha **4**.

Entrada do Programa:

A entrada do programa será um arquivo `.csv` contendo todos os comandos a serem executados pelo jogo. Cada comando consistirá de uma posição inicial (posição da peça a ser movida) e uma posição final (posição para onde a peça selecionada será movida). No arquivo `.csv` os comandos serão separados por vírgulas, ou seja, cada comando, contendo posição inicial e final, será separado por vírgula do próximo. Está sendo disponibilizado uma classe (CSVReader) para ler esse arquivo `.csv` e retornar a entrada pronta num vetor de String, onde cada posição desse vetor consiste da posição inicial e da posição final separadas por ":".

Exemplo de uma posição do vetor contendo as entradas: **f4:d4**. Nesse exemplo, a posição inicial é a coluna **f** e a linha **4** e a posição final é a coluna **d** e a linha **4**.

Quando um peão é promovido para uma outra peça, na entrada subsequente ao movimento haverá a letra da peça para a qual ele será promovido, por exemplo, se um peão branco alcançar a posição **a8** e resolver se transformar em uma rainha, a entrada será assim **a7:a8,q**.

A entrada usará o mesmo CSVReader da etapa anterior do trabalho, entretanto, nesta etapa você deve estender esta classe de tal forma que ela seja capaz de diferenciar o tipo de entrada. Para isso, a classe herdeira deve transformar o vetor de `commands` (originalmente apresentado como vetor de Strings) em um vetor de objetos de comando. O tipo de comando é reconhecido de acordo com a classe do objeto. Comandos da classe `Movimento` representam movimentos e aqueles da classe `Transforma` representam transformação em outra peça. Explore o polimorfismo para que todos os comandos sejam dispostos em um vetor único, seguindo a estratégia feita no `Emprestimo`.

Saída do Programa:

A saída do programa deve ser mostrada no console e segue o mesmo princípio da etapa anterior do trabalho. Deve ser mostrada a posição inicial (source) e final (target) da peça que vai ser movimentada na rodada, bem como o estado do tabuleiro após a movimentação. Antes da primeira movimentação mostre o estado inicial do tabuleiro.


Entrega

A entrega deve ser realizada via Github. Vocês devem submeter o código-fonte no seu Github e enviar, pela atividade do Classroom, o link do seu repositório no Github. A imagem abaixo mostra como enviar um link na atividade:

- 1) Entre na Atividade;
- 2) Clique em “Adicionar ou criar”;
- 3) Link


Seus trabalhos

Pendente



ir-atoms.ipynb

Arquivo binário



+ Adicionar ou criar



Google Drive



Link



Arquivo