Relatório Projeto 1

Caio Fernando Peres - 769298

Neste projeto foi implementado em java, até o momento, um jogo de xadrez de forma parcial.
Ainda falta a classe Peça e todos os métodos que dependam dela, portanto o jogo está incompleto.
Foram utilizados todos os conceitos de orientação à objetos cabíveis até o momento.
Neste documento consta um relatório do desenvolvimento de todas as classes e métodos possíveis até então.

Descrição das classes:

- Classes de peças específicas:
    - O método checaMovimento() é diferente em todas as classes.
    - O método desenho() é o mesmo em todas as classes, apenas trocando a letra que será mostrada com a respectiva primeira letra do nome da Classe. Em maiusculo caso receba 1 como parametro, e minusculo caso receba 2.
    - Os seguintes métodos são iguais em todas as classes:
        - O método getCapturada() retorna o atributo capturada.
        - O método setCapturada() define um valor para o atributo capturada.
        - O método getCor() retorna o atributo cor.
        - O método setCor() define um valor para o atributo cor.

- Rei:
    - Possui atributos inteiros capturada e cor.
    - Possui métodos checaMovimento(), desenho(), getCapturada(), setCapturada(), getCor(), setCor().
    - O método checaMovimento() é específico da classe, pois checa se o movimento compreende em qualquer direção, andando apenas uma casa.
    - Utiliza construtor padrão.

- Dama:
    - Possui atributos inteiros capturada e cor.
    - Possui métodos checaMovimento(), desenho(), getCapturada(), setCapturada(), getCor(), setCor().
    - O método checaMovimento() é específico da classe, pois checa se o movimento compreende em qualquer direção, quantas casas quiser.
    - Utiliza construtor padrão.

- Bispo:
    - Possui atributos inteiros capturada e cor.
    - Possui métodos checaMovimento(), desenho(), getCapturada(), setCapturada(), getCor(), setCor().
    - O método checaMovimento() é específico da classe, pois checa se o movimento compreende nas diagonais, quantas casas quiser.
    - Utiliza construtor padrão.

- Cavalo:
    - Possui atributos inteiros capturada e cor.
    - Possui métodos checaMovimento(), desenho(), getCapturada(), setCapturada(), getCor(), setCor().
    - O método checaMovimento() é específico da classe, pois checa se o movimento compreende em L, com um número específico de casas.
    - Utiliza construtor padrão.

- Torre:
    - Possui atributos inteiros capturada e cor.
    - Possui métodos checaMovimento(), desenho(), getCapturada(), setCapturada(), getCor(), setCor().

    - O método checaMovimento() é especifico da classe, pois checa se o movimento compreende na horizontal e na vertical, quantas casas quiser.
    - Utiliza construtor padrão.

- Peao:
    - Possui atributos inteiros capturada e cor.
    - Possui métodos checaMovimento(), desenho(), getCapturada(), setCapturada(), getCor(), setCor().
    - O método checaMovimento() é especifico da classe, pois checa se o movimento compreende 2 casas apenas na primeira jogada, ou 1 casa no resto do jogo, ou na diagonal (apenas para comer outra peça, mas isso quem checa é o tabuleiro).
    - Utiliza construtor padrão.
    - É o único que utiliza a cor para a checagem do movimento.

- Classes para o funcionamento do jogo:

    - Gerenciador
        - Possui apenas o método main.
        - Inicia o jogo com os nomes dos jogadores, ao instanciar um objeto Jogo, e gerencia as jogadas.

    - Tabuleiro
        - Possui o atributo posicoes do tipo Posicao, em forma de matriz de posições, instanciada com tamanho fixo de 8x8 posições.
        - Possui construtor para instanciar posições para a matriz de posições, bem como inicializar tais posições com linha, coluna (em caracter) e cor.
        - Possui método checaMovimento() que verifica se o movimento é válido, checando, por enquanto, se pertence aos limites do tabuleiro e se a posição de destino é a mesma da origem.
        - Possui método getPosicao() que retorna uma posicao null caso a posição requisitada esteja fora dos limites do tabuleiro ou retorna a posição requisitada caso esteja dentro dos limites.
        - Possui método desenhar() que desenha na tela o tabuleiro atual (por enquanto, sem peças).

- Posicao
    - Possui os atributos inteiros linha e cor, e o atributo de caracter coluna, todos declarados como final.
    - Possui um construtor que inicializará os atributos com o valor recebido.
    - Possui métodos getLinha(), getColuna(), getCor().

- Jogo
    - Possui os atributos jogador1 e jogador2 do tipo Jogador, os atributos inteiros vez e estado, e o atributo tabuleiro do tipo Tabuleiro.
    - Possui um construtor que irá inicializar o tabuleiro com o objeto tabuleiro, e os jogadores com objetos de jogador, vez começando com 1 pois o branco sempre começa no xadrez e estado com 0.
    - Possui o método jogada(), que fará a jogada do jogador a partir do programa principal. Nesse método, é preciso converter as linhas recebidas para índices reais de vetores, uma vez que o tabuleiro não começa com a linha 0. Além disso, também é preciso converter os caracteres recebidos das colunas para valores reais de índice, e nesse processo já é feita a verificação dos valores das colunas. Depois, checa o movimento da peça através do método checaMovimento() do tabuleiro. Por fim, troca a vez do jogador e desenha o tabuleiro no console chamando o método desenhar do tabuleiro.
    - Possui método estático limparTela(), que é uma forma de limpar a tela dos terminais do Windows ou Linux, como forma de refinar a interação com o usuário.
    - Possui método getNomeJogador() que retorna o nome do jogador da vez.
    - Possui método getVez() que retorna a vez atual.
    - Possui método getTabuleiro(), que aparentemente não seria necessário ao jogo finalizado, porém para conseguir testar todas as funcionalidades do jogo foi necessária a sua criação.
    - Possui método getEstado() que retorna o estado atual do jogo.
    - Possui método setEstado() que atribui um estado para o atributo estado da classe.

- Jogador
    - Possui atributo nome do tipo String.
    - Possui um construtor que atribui uma string para o nome.
    - Possui método getNome(), que retorna o nome do jogador.

    - Possui método setNome(), que define um nome para o atributo nome.

Testes:

Foram testadas, a partir de uma classe Teste, todas as classes e métodos criados, e todos os problemas encontrados foram corrigidos. Portanto, considerando os testes feitos até o presente momento, não se conhece condições que possam causar um erro que não está sendo tratado. O tratamento consiste em apresentar uma mensagem de erro para o usuário.
Para facilitar os testes, alguns métodos foram testados a partir do construtor da classe.
Vários testes precisaram ser em conjunto, pois um método dependia de outro, e alguns foram feitos juntos para facilitar.
Das condições que podem causar um erro tratado são:

- Entradas que extrapolam os limites do tabuleiro
- Coluna como número
- Movimento que retorna para a mesma posição

Descrição dos testes:

- Teste 1
    - Objetivo: Verificar as cores, linhas e colunas das posicoes.
    - Objeto: jogoXadrez, tabuleiro(dentro da classe Jogo), posicoes(dentro da classe tabuleiro).
    - Método testado: getTabuleiro() da classe Jogo, getPosicao() da classe Tabuleiro. getCor(), getLinha() e getColuna() da classe Posicao.
    - Valores dos Parâmetros: Somente em getPosicao() foi necessário definir parametros. Os parâmetros definidos variaram dentro de 2 laços de repetição, um para i e outro para j, percorrendo assim todas as posições possíveis.

- Valor de retorno: O método getTabuleiro() retorna o tabuleiro; o método getPosicao() retorna a posição; o método getCor() retorna o inteiro cor; o método getLinha() retorna o inteiro linha e o método getColuna() retorna o caracter coluna.
- Com o teste, foi identificado um erro de sintaxe ao atribuir a cor, pois suas saídas não eram condizentes com o esperado. Ao fazer a verificação se o valor da linha era par ou ímpar, faltou parênteses. Dessa forma, a cor estava sendo atribuída de forma incorreta. Após a correção, a condição ficou: (i+1) % 2 ao invés de i+1 % 2, e o problema foi resolvido.
- Tela: Na tela foi mostrado as variáveis cor, linha e coluna, bem como uma string antes da variável com sua descrição para facilitar a compreensão.

- Teste 2
  - Objetivo: Verificar se o nome do jogador é alterado em cada jogada.
  - Objeto: jogoXadrez
  - Método testado: getNomeJogador() e jogada().
  - Valores dos parâmetros: linhaOrigem, colunaOrigemChar, linhaDestino e colunaDestinoChar como parâmetros para o método jogada().
  - Valor de retorno: O método getNomeJogador() retorna uma string contendo o nome do jogador atual.
  - Para testar se o nome está sendo alterado a cada jogada, é preciso fazer uma jogada. Para fazer a jogada, é preciso fornecer os valores de origem da coluna e linha, e os valores de destino, e passar para o método jogada(). Os valores são fornecidos pelo usuário. Após o teste, foi constatado que o nome se altera a cada jogada, a menos que o jogador insira uma entrada inválida. Neste caso, a vez continua sendo do jogador que inseriu a entrada inválida, para que ele possa fazer uma jogada correta dessa vez. Portanto, o teste foi um sucesso.
  - Tela: É mostrado na tela o nome do jogador atual.

- Teste 3
  - Objetivo: Verificar se é possivel limpar a tela.
  - Objeto: é aplicado à classe Jogo por ser método estático.
  - Método testado: limparTela()
  - Valores dos parâmetros: nenhum.
  - Valor de retorno: nenhum.

- O método chamado limpa o terminal que estiver em execução. Só funciona no terminal do Windows e do Linux. Portanto, para o teste foi necessário rodar o programa compilado no terminal. Após o método ser chamado, o terminal é limpado, concluindo o teste com sucesso.
- Tela: A tela é limpa.

- Teste 4
  - Objetivo: Verificar o método checaMovimento() de todas as peças
  - Objeto: rei, dama, cavalo, bispo, torre, peao
  - Método testado: checaMovimento() de todos os objetos acima.
  - Valores dos parâmetros: linhaOrigem, colunaOrigem, linhaDestino e colunaDestino.
  - Valor de retorno: 0, 1 ou 2.
  - O único método que pode retornar 2 é o método do peao, pois ele necessita de uma verificação a mais do que os outros métodos (só pode andar na diagonal na condição de comer uma peça).
    Esse teste foi inserido dentro de um laço de repetição para que o teste fosse mais fácil de ser feito. Foram testadas várias combinações de movimentos para checar se algum método tinha resultado incorreto. Para este teste funcionar, foi necessário chamar o método setCor() do objeto peao e definir uma cor (0 ou 1). Um teste mais completo foi feito com todas as peças individuais no Teste 7.
    O método do peao tinha um erro em que o peao de cor preta podia voltar posições. Isso de deve a um erro ao fazer a condição em que ficou faltando comparar a cor. Após o teste, o erro foi solucionado.
  - Tela: Nada.

- Teste 5:
  - Objetivo: Verificar tratamento de entradas inválidas.
  - Objeto: jogoXadrez e tabuleiro
  - Método testado: jogada(), checaMovimento() e desenhar().
  - Valores dos parâmetros: linhaOrigem, colunaOrigemChar, linhaDestino, colunaDestinoChar, colunaOrigem e colunaDestino.
  - Valor de retorno: O Método checaMovimento() da classe Tabuleiro retorna 1 ou 0.
  - Ao fazer a jogada, dentro do método jogada() é feita uma conversão de linha e coluna da origem e do destino, para valores de índice. Após a conversão, é feita uma checagem se existe valor inválido de coluna. Caso exista, é feita uma impressão de erro na tela e retornado. Caso

não exista valor inválido de coluna, é então chamado o método checaMovimento() do objeto tabuleiro com os parâmetros linhaOrigem, colunaOrigem, linhaDestino, colunaDestino já convertidos para índice. Dentro do método é feita a verificação se as linhas são válidas. Caso não sejam, é retornado o valor 0 e então feita a impressão da mensagem de erro e retornado. Caso sejam, é retornado 1 e a jogada será feita (falta implementação da classe peça). Após a jogada, a vez é mudada para o outro valor possível e é chamado o método desenhar() do objeto tabuleiro, que desenhará o tabuleiro na tela. Como esse teste também está dentro de um laço de repetição, foi utilizado vários valores de entrada e foi possível perceber que todas as entradas inválidas estão sendo tratadas com mensagens de erro.
    - Tela: Mensagem de erro.

- Teste 6:
    - Objetivo: Verificar o estado
    - Objeto: jogoXadrez
    - Método testado: getEstado()
    - Valores dos parâmetros: nenhum
    - Valor de retorno: estado.
    - O estado é verificado ao ser mostrado na tela, concluindo o teste do método.
    - Tela: É mostrado o estado.

- Teste 7:
    - Objetivo: Verificar os getters, setters e desenho das peças individuais
    - Objeto: rei, dama, cavalo, bispo, torre e peao
    - Método testado: setCapturada(), getCapturada(), setCor(), getCor(), desenho().
    - Valores dos parâmetros: 1 para os setters, 1 para desenho.
    - Valor de retorno: É retornado o capturada e a cor.
    - Ao utilizar os métodos getters, setters e desenho das peças específicas, foi possível notar que todos funcionam corretamente.
    - Tela: Foi mostrado na tela o valor dos atributos capturada e cor, e também o desenho (caracter) das classes através dos métodos.

Diagrama de Classes: