TRABALHO - CI1062

Nome: Erick da Silva Santos – GRR20182554

Link do Replit: https://replit.com/@sserick/TrabalhoCI1062?v=1

1. **Classes (atributos e métodos)**

Utilizei classe com atributos e métodos em quase todo os arquivos, a pasta com funções e funções de geração são classes com códigos que serão utilizados durante o projeto.

O arquivo *Imprimir* dentro da pasta *funcao*, possui algumas funções privadas, que utilizei como forma de aproveitar código, assim como no arquivo *GerarLadosSetor* dentro da pasta *funcoesGeracao.*

2. **Construtores**

Utilizei construtores em todos os arquivos em que realizei herança (Pai e filho) que possuía atributos como no caso do *Setor -> SetorNormal, SetorOculto e SetorPrivado*. Na classe Setor utilizei sobrecarga de construtores, pois na inicialização do setor eu não precisaria utilizar todos os parâmetros, então utilizei a sobrecarga para inicializar somente os atributos do setor que precisava.

3. **Encapsulamento**

Em todas as classes que utilizei atributos, como *Posicao, Setor, GerarTabuleiro, Inimigo, Personagem, JogadorSimples e JogadorSuporte,* fiz a utilização do encapsulamento deixando os atributos com o modificador de acesso Protected e fazendo a utilização dos Getters e Setters, em alguns casos fazendo validações no método Setter.

4. **Herança**

Para algumas classes utilizei herança como forma de aproveitamento de código e facilitar na organização. A herança das classes pode serem visualizadas melhor no diagrama que montei na ultima pagina.

5. **Interface**

Nesse projeto não utilizei interfaces, pois seguindo o modelo de código que utilizei fazia mais sentido utilizar classes abstratas para aproveitando de código, sendo assim o esquema não abrangia uma classe, na qual eu poderia colocar os mesmos métodos onde subclasses teriam suas especificidades, pois a maior parte fazia a mesma coisa, como é o caso do personagem,

inimigo, jogador simples e jogador de suporte tinha o mesmo atributo (está vivo), os três teriam a ação ataque, porém a do jogador é diferente do inimigo.

6. Classe Abstrata

Utilizei classe abstrata somente na classe Personagem pois a maioria das classes possuía métodos diferentes ou aproveitamento de métodos e como a classe abstrata não pode ser instanciada, foi a melhor forma que achei para utiliza-la.

7. Polimorfismo

Na parte de polimorfismo utilizei o polimorfismo de inclusão nos setores, onde definia inicialmente o tabuleiro como Setor[][] e fazia a inserção de acordo com o tipo de setor (setor normal, setor oculto e setor privado).

Utilizei também polimorfismo nos jogadores, pois criei uma classe do tipo jogador para conter todos os atributos e métodos em comum, porem dentro da classe de cada jogados coloquei alguns atributos e métodos que teriam que ser trabalhados diferentes, como no caso do método setAtaque e setDefesa de cada jogador que pode receber número mínimo e máximo diferentes. Nesse caso para acessar essas informações mais específicas utilizei o polimorfismo AdHoc de coerção Down Cast, fazendo um jogador ser convertido para o tipo mais especifico: jogador simples ou jogador de suporte.

8. Coleção

Na coleção utilizei um ArrayList para representar a coleção de cada inimigo dentro do setor, podendo assim fazer modificação pelo índice e quando o inimigo estivesse morto faria a remoção.

Esquema de herança