



Prática – Padrões Simple Factory / Factory Method

a) Utilizando o padrão *simple factory* crie uma fábrica capaz de construir aleatoriamente 5 personagens diferentes, com diferentes ataques, diferentemente decorados. Use o método `random()` da classe `Math` para gerar um número aleatório `N` de 0 a 1 e retornar um personagem da seguinte forma:

Se $N \leq 0.2$ retorna personagem tipo 1

Se $0.2 < N \leq 0.4$ retorna personagem tipo 2

Se $0.4 < N \leq 0.6$ retorna personagem tipo 3

Se $0.6 < N \leq 0.8$ retorna personagem tipo 4

Se $0.8 < N \leq 1.0$ retorna personagem tipo 5

Teste a fábrica no método `main()`.

b) Crie uma *simple factory* capaz de construir aleatoriamente 5 diferentes tipos de personagens avançados (`AdvancedFactory`), com seus ataques decorados por poderes especiais, criados anteriormente com o padrão *decorator*.

c) Com objetivo de tornar o *client* (jogo) capaz de utilizar qualquer uma das fábricas em tempo de execução, aplique o padrão *factory method*, criando uma implementação abstrata da fábrica, mas deixando cada sub-classe concreta decidir quais personagens criar.

Na classe abstrata crie um método `jogar()` que implemente uma lógica de execução do game e utiliza o *factory method* para fabricar personagens e inimigos.

As *simple factories* criadas anteriormente passam agora a implementar os métodos fábricas concretos como sub-classes da fábrica abstrata.
