Gabriel Pellegrino da Silva /172358

Respostas do Laboratório 4

1.

```
Paulo (id: 87779dc6-1089-491a-a7fa-7e1422ac2608)
Custo de Mana = 10
Ataque = 10
Vida Atual = 10
Vida Maxima = 10
Beatriz (id: b6b59913-0d89-4628-9365-ce9ebbc04466)
Custo de Mana = 5
Ataque = 5
Vida Atual = 5
Vida Maxima = 5
Buffer (id: 29eb98c6-660f-472e-9688-80f7c8331cb6)
Custo de Mana = 10
Aumento em Ataque = 20
Aumento em Vida = 30
Dano (id: 0725da2b-d374-4a8b-8998-da74a3915bb9)
Custo de Mana = 2
Dano = 5
Dano Area (id: 614cb68e-773b-4d2c-b120-d0bf7600c907)
Custo de Mana = 10
Dano = 10
```

2.

Nesse exemplo, nao faz sentido instanciar objetos do tipo Carta, pois no baralho, podemos especializar o vetor de cartas em tipos diferentes de cartas, tais como Lacaios e Magias.

3.

Nao, pois a sobrescrita de metodo esta relacionada com a chamada de um metodo com o mesmo nome e a mesma lista de parametros, mas definidos em classes diferentes.
Como nao ha outro metodo com essas caracteristicas na 'familia' da classe DanoArea, nao houve sobrescrita.

4.

Sim, o atributo nome de um Lacaio pode ser acessado atraves do metodo presente na sua superclasse (Carta). Ao invocar o metodo, verifica-se a sua presenca na classe Carta, e depois verificasse verticalmente na heranca,

ate encontrar, ou nao, o metodo.

5.

Metodos estaticos nao sao herdados pelas subclasses.
Mas, uma maneira de se fazer um metodo estatico de classe ser chamado
por uma subclasse eh a seguinte:

```
    ((Carta)lac1).meuMetodoEstatico();
```

Dar um cast na subclasse para ter acesso as propriedades da superclasse.

6.

Os beneficios de heranca sao diminuicao de codigo, encapsulamento e possi
bilidade de expansao do sistema de maneira mais segura: as classes dependem
entre si, mas tambem possuem um comportamento proprio. Ou seja, fazer modi
ficacoes nas classes torna-se muito mais simples.
Alem da possibilidade do polimorfismo do Baralho.

Resposta ao feedback do Laboratorio 3

Nao, como dito pela professora em sala, inverter e desenverter 30 posicoes
na memoria nao eh nada para um computador moderno.
Optei por essa opcao simplesmente por nao conhecer de cara o metodo reverse
da classe Collections e por concluir que em grandes sistemas ele seria mais
agil.

Acredito que legibilidade de codigo eh muito importante, mas isso tambem po
de ficar por conta da documentacao. E durante a escrita do codigo focar em
otimizacao.

Ps: nao sei formular uma documentacao decente xD

Sobre as outras possibilidades de formular o baralho, pensei em fazer
ArrayList's separados para cada tipo de carta e depois concatena-los.
Fora a (classica em C) utilizacao de flag's para marcar o 'tipo' da Struct,
agora o 'tipo' da Classe :D