

Exercícios - Herança

Exercício 1

Os super-heróis são personagens fictícios dotados de poderes sobre-humanos. Criados pela imaginação do homem, eles estão sempre em alerta para proteger o mundo dos ataques de mentes cruéis que pretendem dominar o nosso planeta.


Cada super-herói tem uma origem interessante. Alguns, como o Incrível Hulk, o Capitão América e o Homem de Ferro, surgiram em laboratórios, e eram pessoas comuns antes de adquirirem seus superpoderes a partir de acidentes ou experiências com raios-gama, reações químicas e estudos científicos nos campos da física, engenharia e biologia. Outros, como o Super-Homem e o Lanterna-Verde, vieram de outros planetas. Existem ainda aqueles que se originaram da mitologia grega, romana ou nórdica, como é o caso do Thor e a Mulher-Maravilha.

Os primeiros super-heróis apareceram entre 1930 e 1960, e o desenho em quadrinhos foi o principal veículo de divulgação em massa, antes da chegada da televisão. As principais empresas do ramo são a Hanna-Barbera, produtora da afamada Liga da Justiça (Super-amigos, 1973) e a Marvel, criadora de dezenas de personagens "vivos" até hoje.






Em quase todas as histórias inventadas, o super-herói é chamado para resolver um problema ou enfrentar ameaças de um vilão com um plano maligno. O vilão também é munido de superpoderes mas dificilmente consegue vencer o super-herói, pois os poderes deste são mais fortes.

Na lista abaixo são citados alguns super-heróis e alguns vilões, dos quais indubitavelmente já ouvimos falar. A lista apresenta também o nome na vida real e os superpoderes de cada um. Os superpoderes foram categorizados de 1 a 5, sendo 5 o poder mais forte e 1 o poder mais fraco.

Super-Heróis

Nome	Nome na vida real	Superpoderes	Categoria do Poder
Homem-Aranha 	Peter Park	soltar teia andar em paredes sentido apurado	3 2 1




Exercícios - Herança

Super-Homem 	Clark Kent	voar força visão de raio X sopro congelante	3 5 4 4
Capitão América 	Steven Rogers	supersoldado escudo	3 3
Flash 	Barry Allen	velocidade	5
Lanterna-Verde 	Hal Jordan	anel mágico	5
Homem de Ferro 	Tony Stark	armadura dispositivos eletrônicos	4 2

Vilões:

Duende Verde 	Norman Osbourne	Força	5
---	-----------------	-------	---

Exercícios - Herança

Lex Luthor 	Lex Luthor	Mente aguçada	5
Bizarro 	Bizarro	voar força visão de raio X sopro congelante	3 5 4 4
Octopus 	Otto Octavius	tentáculos indestrutíveis velocidade	5 1

1. Construa uma classe chamada **Superpoder**, com a seguinte estrutura:

- Atributos privados
 - onome: String
 - ocategoria: int
- Métodos públicos
 - ogetNome()
 - retorna o nome do poder
 - ogetCategoria()
 - retorna a categoria do poder
- Construtor público
 - oSuperpoder(String nome, int categoria)
 - Recebe o nome do poder e a categoria e atribui ao objeto.

2. Construa uma classe denominada **Personagem**, com a seguinte estrutura:

- Atributos privados
 - onome: String
 - onomeVidaReal: String
 - opoderes: vetor de 4 elementos do tipo Superpoder
- Métodos públicos
 - ovoid adicionaSuperpoder(Superpoder sp)
 - recebe um superpoder como parâmetro e coloca-o no vetor 'poderes'. Um super-herói ou vilão pode ter, no máximo, 4 poderes.

Exercícios - Herança

```
int getPoderTotal()
    retorna a soma de poderes do super-herói. O poder total é
    calculado percorrendo-se o vetor 'poderes' e somando-se a
    categoria de cada poder.
```

▪ Construtor público

```
oPersonagem (String nome, String nomeVidaReal)
    Recebe os nomes do personagem e atribui ao objeto.
```

3. Construa uma classe chamada **SuperHeroi**, que descende da classe **Personagem**, com a seguinte estrutura:

▪ Construtor público

```
oSuperHeroi(String nome, String nomeVidaReal)
    Recebe os dois parâmetros e repassa para a classe base,
    Personagem.
```

▪ Redefinição do método da classe base

```
int getPoderTotal()
    Acrescenta um inflator de 10% aos poderes do personagem
    super-herói.
```

4. Construa uma classe chamada **Vilao**, que descende da classe **Personagem**, com a seguinte estrutura:

▪ Atributos

```
otempoDePrisao: int
```

▪ Construtor

```
oVilao(String nome, String nomeVidaReal, int tempoDePrisao)
    Recebe os três parâmetros e repassa dois deles para a
    classe base, Personagem.
```

5. Construa uma classe chamada **Confronto**, com a seguinte estrutura:

▪ Métodos públicos

```
int executar (SuperHeroi superheroi, Vilao vilao)
    o método recebe um super-herói e um vilão como parâmetros
    e decide quem é o vencedor da batalha. O método deve
    retornar:
```

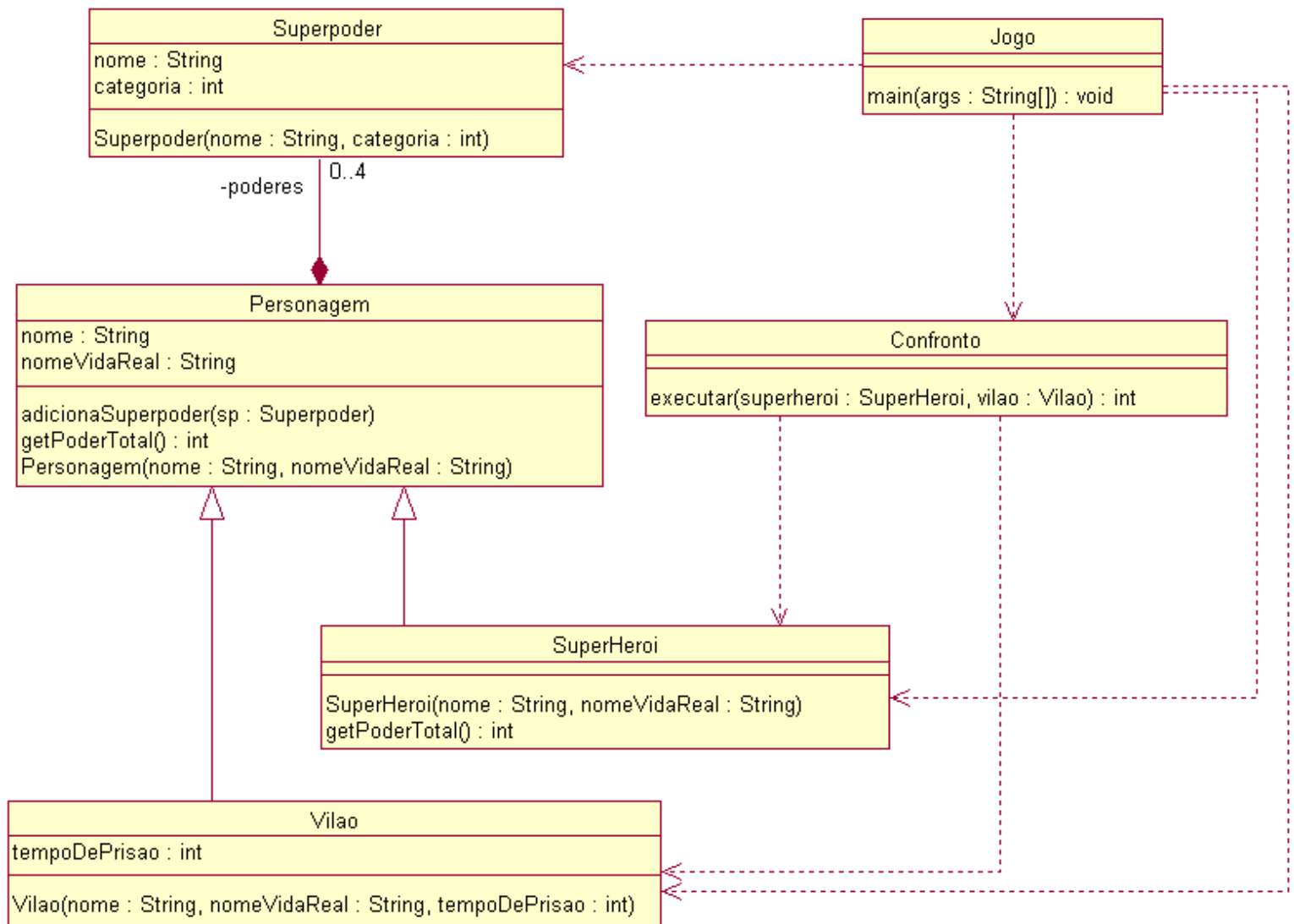
- 1 se o super-herói ganha a batalha
- 2 se o vilão ganha a batalha
- 0 se houver empate.

O vencedor será aquele que tiver mais poder (use o método `getPoderTotal()` para saber qual é o poder de cada um).

6. Construa uma classe chamada **Jogo**, com a função **main**. Nessa função, faça o seguinte:

- a) crie um objeto da classe **SuperHeroi** e um objeto da classe **Vilao**.
- b) crie os superpoderes e atribua-os aos objetos **SuperHeroi** e **Vilao**.
- c) crie um objeto da classe **Confronto** e invoque o método `executar`, passando como parâmetro o super-herói e o vilão.
- d) Mostre uma mensagem na tela dizendo quem é que vence o confronto.

Exercícios - Herança



Exercícios - Herança

Exercício 2

Um polígono regular é um polígono que possui todos os lados com o mesmo comprimento. Assim, um quadrado e um triângulo equilátero são polígonos regulares. Neste exercício, crie uma classe base para representar polígono regular e outras duas classes para tratar quadrados e triângulos. Cada polígono deve saber como calcular sua área. Faça uma quarta classe, que instancia 5 polígonos (2 quadrados e 3 triângulos) e calcula a área total.

a. Área do quadrado: $lado^2$

b. Área do triângulo equilátero: $\frac{1}{4} lado^2 * \sqrt{3}$

Exercício 3

Desenvolva em Java as classes para representar o seguinte cenário, baseando-se no diagrama de classes informado no final deste exercício.

1. Uma dança é um tipo de habilidade artística. Essa habilidade é executada por pessoas (artistas dançarinos) por meio de uma série de movimentos encadeados. Cada movimento executado pode ser tão simples como um passo para frente, ou então visto como um movimento composto. Neste caso, ele pode ser fracionado sucessivamente em uma série de movimentos menores até chegar a um movimento simples.
2. Um artista pode ter mais de uma habilidade, como cantar ou representar, etc.
3. Os artistas e as pessoas (público) comparecem ao teatro com propósitos diferentes. Uma vez dentro do auditório, todos esses tipos de pessoas ficam atentos para saber o que devem fazer. Quando a coordenação do teatro sinaliza o início da apresentação, a ação que os artistas tomam é, obviamente, executar sua(s) habilidade(s) para o público. Este, por sua vez, toma a ação de ficar em silêncio e apreciar o espetáculo. Quando a apresentação é sinalizada como encerrada, o público toma outra ação, que é aplaudir ao espetáculo, enquanto os artistas tomam a ação de agradecer.
4. No método main da classe Coordenacao, implemente a instanciação do teatro, das pessoas (público e artistas, estes com algumas habilidades artísticas). Em seguida, registre a entrada das pessoas no teatro, inicie o espetáculo e depois finalize-o.
5. Os métodos de iniciar espetáculo e finalizar espetáculo na classe Teatro devem modificar o estado do Teatro (espetáculo em execução, espetáculo finalizado).
6. Como o teatro passa a "conhecer" as pessoas registradas na entrada (cada objeto Pessoa fica armazenado na lista de pessoas da classe Teatro), ele pode notificar cada uma delas invocando o método realizeUmaAcaoNoTeatro e passando a si mesmo como argumento do método.
7. Para simplificar, o método realizeUmaAcaoNoTeatro da Pessoa pode ser implementado com uma saída qualquer (System.out.println...), conforme a situação do Teatro (espetáculo em execução, espetáculo finalizado). Entretanto, este método está sobrescrito na classe Artista. Quando invocado, deve-se percorrer todas as habilidades adicionadas ao Artista e invocar o método executeHabilidade de cada uma delas.
8. No caso da habilidade Dança, a execução da habilidade é delegada ao Movimento, invocando-se o método executeMovimentos.

Exercícios - Herança

