Java Polimorfismo: entenda herança e interfaces

⭐ Questão 1 (Java)

Das alternativas abaixo, quais são verdadeiras sobre construtores em Java?

Resposta:

- O construtor padrão (default) é aquele que não recebe nenhum parâmetro.
- O construtor é chamado na inicialização do objeto.

⭐ Questão 2 (Java)

Quais foram os problemas apresentados na implementação da classe Funcionario?

Respostas:

- Repetição de código;
- Muitos ifs que nunca param de crescer;
- Código pouco expressivo.

⭐ Questão 3 (Java)

Vimos como estender uma classe no Java, por exemplo:

```
class Gerente extends Funcionario {}
```

A classe Gerente, ao estender a classe Funcionario:

> A) Herda todas as características da classe Funcionario.
> B) Herda todo o comportamento da classe Funcionario.
> C) Instancia um Funcionario.
> D) É um Funcionario.
> **Quais das afirmativas são verdadeiras?**

Resposta:

- Afirmativas A, B e D.

⭐ Questão 4 (Java)

Qual é a sintaxe correta para estender uma classe no Java?

Resposta:

```
class Carro extends Veiculo {}
```

⭐ Questão 5 (Java)

Em relação ao que você aprendeu até agora, qual é a ordem correta dos modificadores de visibilidade, da menor visibilidade para a maior?

Resposta:

- private < protected < public

⭐ Questão 6 (Java)

Vimos que a sobrescrita é um conceito importante na herança, pois permite redefinir um comportamento previsto na classe mãe através da classe filha. Agora veja a classe Veiculo abaixo:

```
class Veiculo {
    public void liga() {
        // alguma implementação
    }
}
```

E a classe filha Carro:

```
class Carro extends Veiculo {
    // ????
}
```

Qual dos métodos abaixo inserido no lugar de // ???? sobrescreve corretamente o método liga?

Resposta:

```
public void liga() {
    // implementação
}
```

> **Comentário da Alura:** Como regra de ouro, na sobrescrita a visibilidade não pode diminuir, deve ser a mesma ou maior! Repare que o método possui a mesma assinatura. Isto é, a mesma visibilidade, mesmo retorno, mesmo nome e os mesmos parâmetros.

⭐ Questão 7 (Java)

Sobre herança em Java, julgue as seguintes afirmativas:

1. Uma classe pode ter várias filhas, mas apenas uma mãe.
2. A partir de uma instância de uma classe filha, podemos chamar qualquer método público que tenha sido declarado na classe mãe.
3. Na classe filha, podemos escolher o que herdar da classe mãe.
4. No exemplo abaixo, Cachorro também herda tudo da classe Animal: ```java class Animal { // atributos e métodos }

class Mamifero extends Animal { // atributos e métodos }

class Cachorro extends Mamifero { // atributos e métodos } ``` **Quais afirmativas estão corretas?**

Resposta: Apenas as afirmativas 1, 2 e 4 estão corretas.

> **Comentário da Alura:** Pode-se sim chamar qualquer método da classe mãe. Uma classe pode ter diversas "filhas e netas" (que herdam umas das outras), mas não podemos escolher o que será herdado.

⭐ Questão 8 (Java)

Qual é a diferença entre private e protected?

Resposta: Só a própria classe enxerga atributos/métodos private, enquanto protected é visto pela própria classe mais as classes filhas.

⭐ Questão 9 (Java)

Dada a classe Veiculo:

```
public class Veiculo {
    public void liga() {
        System.out.println("Ligando Veiculo");
    }
}
```

A classe Carro:

```
class Carro extends Veiculo {
    public void liga() {
        System.out.println("Ligando Carro");
    }
}
```

E a classe Moto:

```
class Moto extends Veiculo {
    public void liga() {
        System.out.println("Ligando Moto");
    }
}
```

Veja o código com o método main:

```
public class Teste {

    public static void main(String[] args) {

        Veiculo m = new Moto();
        m.liga();

        Veiculo c = new Carro();
        c.liga();
    }

}
```

Ao executar, o que será impresso no console?

Resposta: Ligando Moto Ligando Carro

> **Comentário da Alura:** Sempre será chamado o método mais específico.

⭐ Questão 10 (Java)

Veja o código abaixo, que deve estar dentro do método main:

```
Funcionario f = new Gerente();
f.autentica(1234);
```

Baseado no que você aprendeu na aula, por que o código não compilou?

Resposta: Porque a referência f é do tipo Funcionario e a classe Funcionario não tem o método autentica.

⭐ Questão 11 (Java)

Continuando com o exemplo Veiculo, Moto e Carro:

```
public class Veiculo {
    public void liga() {
        System.out.println("Ligando Veiculo");
    }
}

public class Carro extends Veiculo {
    public void liga() {
        System.out.println("Ligando Carro");
    }
}

public class Moto extends Veiculo {
    public void liga() {
        System.out.println("Ligando Moto");
    }
}
```

E veja o código quase completo:

```
public class Teste {

    public static void main(String[] args) {

        ???? v = new Carro();

    }

}
```

O que podemos inserir no lugar de ???? para compilar o código sem erros?

Resposta: Os tipos Carro ou Veiculo.

⭐ Questão 12 (Java)

No mundo orientado a objetos, o polimorfismo permite que ...

Resposta: Referências de tipos de classes mais genéricas referenciem objetos mais específicos.

⭐ Questão 13 (Java)

Sobre a herança de classes, todas as afirmativas abaixo são verdadeiras, exceto:

Resposta: Quando uma classe herda de outra, ela recebe também seus construtores automaticamente.

> **Comentário da Alura:** Não é verdade, pois recebe apenas seus métodos e atributos. Lembra-se não tem herança de construtores

⭐ Questão 14 (Java)

Aprendemos que a construção de um objeto é baseada em seu(s) construtor(es). Qual das alternativas abaixo é a correta?

Resposta: O construtor default do java deixa de existir a partir do momento que algum é declarado na classe.

⭐ Questão 15 (Java)

Na última aula vimos sobre a anotação @Override. Qual a finalidade dela?

Resposta: É usada para sobrescrever o método da classe mãe, indicando que o método original foi alterado.

⭐ Questão 16 (Java)

Qual das afirmativas abaixo é VERDADEIRA sobre classes abstratas?

Resposta: Não podem ser instanciadas. Para instanciar, devemos criar primeiro uma classe filha não abstrata.

⭐ Questão 17 (Java)

Qual das afirmativas abaixo é verdadeira sobre os métodos abstratos?

Resposta: Não possuem corpo (implementação), apenas definem a assinatura.

⭐ Questão 18 (Java)

Sobre classes e métodos abstratos, das afirmativas abaixo, qual delas é FALSA?

Resposta: Classes e métodos abstratos consomem menos memória e por conta disso melhoram o desempenho do nosso programa.

> **Comentário da Alura:** Essa afirmação realmente é errada. Classes e métodos abstratos não tem relação direta com consumo de memória.

⭐ Questão 19 (Java)

O que é verdade sobre classes abstratas? Selecione todas as afirmações verdadeiras:

Respostas:

- Podem ter atributos
- Podem ter métodos concretos (com implementação)
- Podem ter métodos abstratos (sem implementação)
- Não podem ser instanciadas

⭐ Questão 20 (Java)

Vimos na última aula que não existe herança múltipla em Java. Como podemos contornar a falta disso?

Resposta: Podemos contornar esta situação com o uso de interfaces.

⭐ Questão 21 (Java)

Sobre interfaces, qual das alternativas abaixo é VERDADEIRA?

Resposta: Ela é um contrato onde quem assina se responsabiliza por implementar esses métodos (cumprir o contrato)

⭐ Questão 22 (Java)

Sobre classes abstratas e interfaces, selecione todas as afirmativas verdadeiras:

Respostas:

- Podemos estender apenas uma classe abstrata, mas podemos implementar várias interfaces.
- Todos os métodos de uma interface são abstratos, os de uma classe abstrata podem não ser.

⭐ Questão 23 (Java)

Quanto ao conceito do Polimorfismo marque as alternativas corretas:

Respostas:

- É a capacidade de um objeto ser referenciado por vários tipos.
- Temos polimorfismo quando uma classe extende de outra ou também quando uma classe implementa uma interface.

⭐ Questão 24 (Java)

Como vimos durante o curso e revisamos durante este capítulo, quais das afirmativas abaixo descreve uma vantagem do uso de herança?

Resposta: A herança captura o que é comum e isola aquilo que é diferente entre classes.

⭐ Questão 25 (Java)

Como vimos durante o curso e revisamos durante este capítulo, quais das afirmativas abaixo descreve uma vantagem do uso de interfaces?

Resposta: Garante que todos os métodos de classes que implementam uma interface possam ser chamados com segurança.

⭐ Questão 26 (Java)

Qual das afirmativas a seguir representa uma vantagem do uso de composição e interfaces sobre o uso de herança?

Resposta: Com composições e interfaces teremos mais flexibilidade com nosso código, já que não nos prenderemos ao acoplamento que a herança propõe.