Prova de Programação Orientada a Objetos (POO) em Java

Instruções: A prova contém questões objetivas e subjetivas. Responda com justificativas e exemplos de código quando necessário. Leia atentamente cada questão antes de responder.

---

1. Conceito de Classe, Objeto e Instanciação

1. (V/F) Em Java, uma classe pode existir sem que nenhum objeto dela seja instanciado.
2. (V/F) Todo objeto em Java deve ser criado com a palavra-chave `new`.
3. O que falta no código abaixo para que o objeto seja corretamente instanciado?

```java
public class Pessoa {
    String nome;
    int idade;
}

public class Teste {
    public static void main(String[] args) {
        Pessoa p;
        p.nome = "João";
        p.idade = 30;
    }
}
```

4. Qual a saída do código abaixo?

```java
class Exemplo {
    int x = 10;
}

public class Teste {
    public static void main(String[] args) {
        Exemplo e1 = new Exemplo();
        Exemplo e2 = e1;
        e2.x = 20;
        System.out.println(e1.x);
    }
}
```

5. Escreva uma classe `Carro` com os atributos `marca`, `modelo` e `ano`. Em seguida, crie um objeto dessa classe e atribua valores a seus atributos.
6. Explique a diferença entre uma variável de instância e uma variável `static`.
7. O que acontece se um objeto não for mais referenciado em Java? Como o Java lida com isso?

---

2. Atributos e Métodos de Classe e de Instância

1. (V/F) Métodos `static` podem acessar diretamente atributos de instância.
2. (V/F) Um atributo `final` pode ter seu valor modificado após a inicialização.
3. Identifique e corrija o erro no código abaixo:

```java
class Exemplo {
    static int contador;

    public void incrementar() {
        contador++;
    }
}

public class Teste {
    public static void main(String[] args) {
        Exemplo e1 = new Exemplo();
        Exemplo e2 = new Exemplo();
        e1.incrementar();
        System.out.println(e2.contador);
    }
}
```

4. O que será impresso ao executar o código abaixo?

```java
class Exemplo {
    static int x = 5;
}

public class Teste {
    public static void main(String[] args) {
        Exemplo e1 = new Exemplo();
        Exemplo e2 = new Exemplo();
        e1.x = 10;
        System.out.println(e2.x);
    }
}
```

5. Explique a principal diferença entre métodos `static` e de instância.
6. O que acontece se tentarmos acessar um método de instância dentro de um método `static` sem criar um objeto?

7. Escreva uma classe `Calculadora` com um método `static` para somar dois números e um método de instância para multiplicar dois números.

---

3. Construtores, Sobrecarga e Encapsulamento

1. (V/F) Uma classe pode ter mais de um construtor em Java.
2. (V/F) Se nenhum construtor for definido, Java cria automaticamente um construtor padrão.
3. O que falta no código abaixo?

```
class Produto {
    String nome;
    double preco;

    Produto(String n, double p) {
        nome = n;
    }
}
```

4. Explique a importância dos métodos `getters` e `setters` no encapsulamento.
5. Qual a saída do código abaixo?

```
class Exemplo {
    private int valor;

    public Exemplo(int v) {
        valor = v;
    }

    public int getValor() {
        return valor;
    }
}

public class Teste {
    public static void main(String[] args) {
        Exemplo e = new Exemplo(10);
        System.out.println(e.getValor());
    }
}
```

6. Escreva um código demonstrando a sobrecarga de construtores em uma classe `Pessoa`.
7. Como a palavra-chave `this` pode ser utilizada dentro de um construtor?

---

4. Herança, Abstração e Polimorfismo

1. (V/F) Uma classe abstrata pode ter métodos concretos.
2. (V/F) É possível instanciar diretamente uma classe abstrata.
3. Identifique e corrija o erro no código abaixo:

```java
abstract class Animal {
    void fazerSom();
}
```

4. Qual a saída do código abaixo?

```java
class A {
    void metodo() {
        System.out.println("A");
    }
}
class B extends A {
    void metodo() {
        System.out.println("B");
    }
}
public class Teste {
    public static void main(String[] args) {
        A obj = new B();
        obj.metodo();
    }
}
```

5. Explique a diferença entre sobrecarga e sobrescrita de métodos.
6. Escreva um exemplo de polimorfismo onde um método `fazerSom()` seja implementado de formas diferentes em classes `Cachorro` e `Gato`.
7. Como o polimorfismo ajuda na reutilização de código?

---

5. Interface, Métodos Default e Padrão DAO/JDBC

1. (V/F) Interfaces podem conter métodos com implementação em Java.
2. (V/F) Métodos `default` em interfaces podem ser sobrescritos pelas classes que as implementam.
3. Identifique e corrija o erro no código abaixo:

```java
interface Exemplo {
    int x;
}
```

4. Qual a saída do código abaixo?

```java
interface A {
    default void metodo() {
        System.out.println("A");
    }
}
class B implements A {
}
public class Teste {
    public static void main(String[] args) {
        B obj = new B();
        obj.metodo();
    }
}
```

5. Explique o padrão DAO e sua importância na persistência de dados.
6. Quais os principais passos para estabelecer uma conexão JDBC?
7. Escreva um código simples demonstrando o uso de JDBC para recuperar dados de um banco de dados.

FIM DA PROVA