

Lista de Exercícios

Curso	Ciência da Computação
Disciplina	Linguagem de Programação III
Professor(es)	Otacílio José Pereira
Assunto	

- Leituras recomendadas:

- Livro Deitel: Cap 10 – Prog. OO – Polimorfismo e interface
- Apostila sugerida: Cap 09 – Herança, Reescrita e Polimorfismo

Parte 1 – Ênfase em Herança

Exercício 1) Explique com suas palavras o que é herança na Orientação a Objetos.

Exercício 2) Em uma empresa existem os funcionários, especificamente o funcionário em geral de uma loja (superclasse) e os gerentes e vendedores (subclasses).

- Implemente a superclasse FUNCIONARIO que possui o campo Salário Fixo referente à base de seu salário.
- Implemente as subclasses GERENTE e VENDEDOR. A classe vendedor apresenta o atributo volume de vendas em reais e a classe gerente apresenta o atributo tamanho da equipe que diz respeito a quantos vendedores existem na equipe.
- Implemente o método CalcularSalario que é a soma do Salário Fixo (classe FUNCIONARIO) e a bonificação variável. O gerente tem um acréscimo de 1% do salário para cada vendedor sob sua gerência e o vendedor recebe 2% a mais de bonificação a cada 2000 reais em compras.

Obs.: Na implementação do método, imprima uma mensagem que indica qual o método de qual objeto está sendo invocado.

- Crie uma classe principal e teste as classes e métodos criados.

Exercício 3) Crie ou pense em uma implementação de uma classe Ponto2D com as coordenadas x e y. Depois crie (ou pense) a classe ponto 3D que apresenta também uma coordenada z.

- Quais informações de Ponto2D são aproveitadas em Ponto3D?
- Imagine uma função CalcularDistanciaAteOrigem, com ela seria implementada em Ponto2D?
- Ao implementar a mesma função em Ponto3D seria importante usar a anotação (annotation) @Override. Por que?

Exercício 4) Observe o código a seguir sobre as classes PessoaJuridica, Pessoa e Microempreendedor.

<pre> 9 public class PessoaJuridica 10 { 11 double faturamento = 1000.00; 12 PessoaJuridica() 13 { 14 super(); 15 System.out.println("B"); 16 System.out.println(faturamento); 17 faturamento = 2000.00; 18 } 19 } </pre>	<pre> 12 public class Pessoa { 13 Pessoa() 14 { 15 System.out.println("C"); 16 } 17 } </pre> <pre> 7 public class MicroEmpreendedor 8 extends PessoaJuridica{ 9 MicroEmpreendedor() 10 { 11 super(); 12 System.out.println("A"); 13 System.out.println(faturamento); 14 } 15 } </pre>
---	---

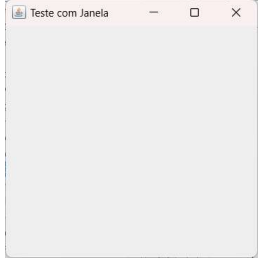
Em uma classe Principal que utiliza estas classes, o que será impresso quando a seguinte linha de código for executada:

Pessoa m = new MicroEmpreendedor();

Exercício 5) Por que a herança é um mecanismo tão importante para o desenvolvimento de software?

Exercício 6) Para deixar a reflexão do exercício 3 ainda mais concreta, observe o código a seguir, ele cria uma janela para o usuário. Seguem os comentários sobre a implementação:

- Perceba que Janela herda de JFrame (classe janela “padrão” no Java) – Linha 9
- A própria janela pode ser executada pois tem o método main – Linha 10
- O novo objeto janela é criado – Linha 12
- E suas características e métodos são acionadas – Linhas de 13 a 15

<pre> 7 import javax.swing.JFrame; 8 9 public class Janela extends JFrame{ 10 public static void main(String[] args) 11 { 12 Janela j = new Janela(); 13 j.setTitle("Teste com Janela"); 14 j.setBounds(100, 100, 300, 300); 15 j.setVisible(true); 16 } 17 } </pre>	
--	--

Ainda que seja uma janela vazia, ao executar você vai perceber que uma janela é criada com todas as funcionalidades de mover, minimizar, maximizar, fechar e outros.

Você precisou de se preocupar com estas funcionalidades? Onde elas foram implementadas?

Parte 2 – Ênfase em Polimorfismo

Exercício 1) De forma mais geral:

- O que você entende por polimorfismo?
- Imagine um cenário com as classes Meio de Transporte, Carro, Lancha e Avião. Conceitualmente, como a ideia de polimorfismo se manifesta? Quais comportamentos (métodos) mais genéricos (Classe Meio de Transporte)? Como este comportamento se concretiza nas várias formas de ocorrer (dentro das classes Carro, Lancha e Avião)?

Exercício 2) Observe o código a seguir. Quais partes (métodos) implementam o polimorfismo? Qual tipo de polimorfismo em questão?

<pre>public class Conta { protected String titular; protected int agencia; protected int numero; protected double valor = 100.0; Conta () { System.out.println("Conta1"); titular = "Novo titular"; agencia = 1; numero = 1; valor = 200.0; } Conta(String pNomeTitular, int pAgencia, int pNumero, double pValor) { System.out.println("Conta2"); titular = pNomeTitular; agencia = pAgencia; numero = pNumero; valor = pValor; } }</pre>	<pre>public void Sacar(Double pValor) { if (pValor < this.valor) this.valor = this.valor - pValor; } public void Sacar(Double pValor, String pMoeda) { if (pMoeda.equals("Dolar")) if ((pValor * 4.15) < this.valor) this.valor = this.valor - pValor; else if (pValor < this.valor) this.valor = this.valor - pValor; }</pre>
--	--

Exercício 3) Em um sistema de geoprocessamento, existem elementos que podem ser pontos (o endereço da sede de uma fazenda), podem ser linhas (as estradas por exemplo) ou podem ser polígonos (as lavouras). Cada elemento possui as coordenadas latitude e longitude de referência. Apresente um exemplo de método geral de todos elementos e um método que pertence apenas a polígono.

Exercício 4) Relembre os códigos das classes Figura2D, Retangulo e Circulo que fizemos em sala. Em relação ao polimorfismo tratado com o método CalcularArea, é um polimorfismo via sobrecarga de métodos ou via sobreposição? Explique.

Exercício 5) Observe o código a seguir. Quais tipos de polimorfismo estão manifestados.

```
public static void main(String[] args)
{
    Conta[] contas = new Conta[5];

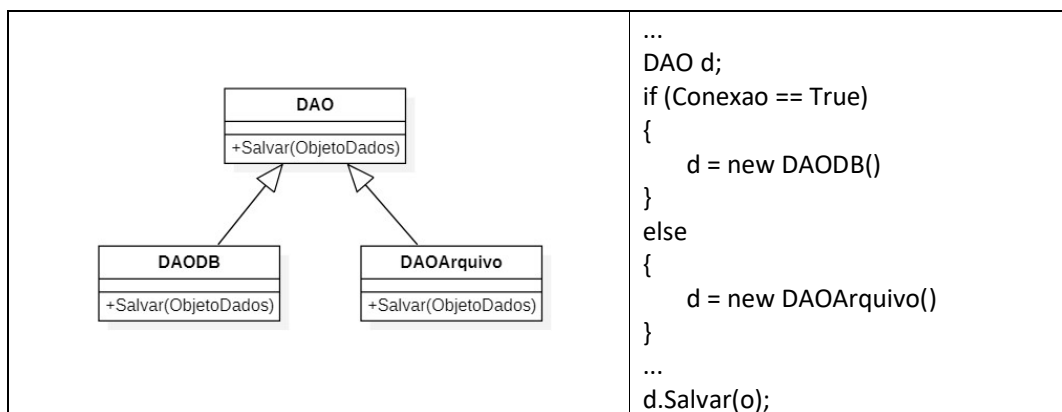
    contas[0] = new Conta();
    contas[1] = new Conta("Fabiana", 10, 20, 0.0);
    contas[2] = new ContaCorrente("Ota", 12, 20, 100.0, 200.0);
    contas[3] = new ContaPoupanca();
    contas[4] = new ContaCorrente();

    for (Conta c : contas)
    {
        c.imprimir();
    }
}
```

Exercício 6) Ao estudar polimorfismo, concluímos um conjunto de princípios que sustentam o paradigma de Orientação a Objetos: Abstração, Encapsulamento, Herança e Polimorfismo.

- Para cada um, apresente exemplos conceituais citando classes, atributos e métodos e outros mecanismos de implementação que fazem com que o Java atenda os princípios.
- Explique como o bom uso destes princípios podem ajudar em um bom desenvolvimento de software.

Exercício 7) Uma aplicação precisa salvar os dados em meio persistente, por exemplo, em arquivo ou em uma Banco de Dados. Para isso foi usado o esquema a seguir, a classe pai é o DAO de Data Access Object (Objeto de Acesso a Dados) e as outras classes herdam de DAO. Observe também o pseudocódigo a seguir sobre salvar um objeto, ele verifica se uma conexão está ativa, daí salva em banco de dados ou em arquivo.



- Com base no entendimento desta solução, avalie qual a vantagem de usar o polimorfismo.
- Caso o sistema agora fosse escrito para salvar os dados em XML ou em outra forma de armazenamento, o que seria necessário fazer. As outras partes do sistema além do DAO precisariam ser alteradas?

Exercício 6) Analise as sentenças a seguir se são verdadeiras ou falsas.

- a) () Em um polimorfismo de sobrecarga métodos pertencem à uma mesma classe mas com nomes diferentes
- b) () No polimorfismo de sobreposição, a distinção entre o método que deverá ser executado ocorre por conta do objeto que recebe o método.
- c) () O polimorfismo de sobrecarga é também considerado um polimorfismo estático.
- d) () O polimorfismo de sobreposição, no caso um tipo de polimorfismo dinâmico, só pode ocorrer entre uma classe abstrata e suas subclasses
- e) () Em um polimorfismo de sobrecarga, a distinção do método a ser executado ocorre por conta da assinatura do método, por exemplo, pelo número e tipos dos parâmetros.
- f) () No nosso cenário de conta, os métodos de Sacar com um parâmetro de valor e Sacar com um valor e tipo de moeda, são exemplo de um polimorfismo de sobrecarga ou estático
- g) () No cenário de uma FiguraGeometrica que pode ser um Triângulo ou Quadrado com um método polimórfico CalcularArea é um exemplo de polimorfismo de sobreposição ou dinâmico

Exercício 7) O que você entende por assinatura de um método? Ele é mais decisivo para o polimorfismo por sobrecarga ou por sobreposição?

Exercício 8) Analise a implementação a seguir. Qual o problema na implementação do polimorfismo?

```
@ public class Conta implements Pix{
12
13     protected double saldo;
14
15     public double getSaldo()
16     {...3 lines }
19
20     public void setSaldo(double pSaldo)
21     {...3 lines }
24
25     public double Sacar(double pValor)
26     {
27         System.out.println(" Classe Conta : Método Sacar(valor)");
28         this.saldo = this.saldo - pValor;
29         return this.saldo;
30     }
```

```
11     public class ContaPoupanca extends Conta{
12
13         public double rendimento;
14
15
16         public double Sacar(int pValor)
17         {
18             System.out.println(" Classe ContaPoupanca : Método Sacar(valor)");
19             this.saldo = this.saldo - pValor;
20             this.rendimento = 0;
21             return this.saldo;
22         }
```

```
11 public class ContaCorrente extends Conta{
12
13     public double limite;
14
15     @Override
16     public double sacar(double pValor)
17     {
18         System.out.println(" Classe ContaCorrente : Método Sacar(valor)");
19         this.saldo = this.saldo - pValor;
20         return this.saldo;
21     }
```

Parte 3 – Ênfase em Classes Abstratas e Interfaces

Exercício 1) Imagine um cenário em que existem as classes Veículo, Carro e Moto.

- a. Explique como o polimorfismo ocorre neste caso e dois exemplos de métodos polimórfico.
- b. Qual classe deve ser abstrata? E dos métodos indicados, qual(is) seria(m) abstrato(s)?

Exercício 2) O que você entende por classe abstrata? E por método abstrato? Apresente dois cenários de exemplo.

Exercício 3) Se em uma classe abstrata com seus métodos abstratos não codifica estes métodos, por que então usá-las? Use um dos cenários acima para ilustrar sua argumentação.

Exercício 4) Imagine um sistema em que as figuras geométricas quadrado, triângulo e círculo representam elementos em um mapa geográfico. Cada elemento pode receber e enviar mensagens, pois podem ser um equipamento instalado na cidade. Com este breve cenário, considerando as classes FiguraGeometrica, Triangulo, Quadrado, Círculo e EquipamentoMensagem, responda:

- a) Qual(is) deve(m) ser classe(s) abstrata(s)?
- b) Qual(is) deve(m) ser classe(s) concreta (s)?
- c) Qual(is) deve(m) ser interfaces(s)?
- d) Explique suas decisões anteriores?
- e) Apresente pelo menos dois exemplos de métodos abstratos das classes?
- f) Apresente pelo menos dois métodos da interface.

Exercício 5) Imagine uma classe usuário que pode ser autenticado de algumas formas, ou por Login e Senha, ou por Biometrica (uma string que representa a biometria) ou por um Inteiro (capturado do chip de um cartão) e a senha. Escreva pelo menos as assinaturas dos três métodos aplicando um polimorfismo de sobrecarga.

Exercício 6) Analise as sentenças a seguir a respeito de classes abstratas e interfaces.

- a) () Nas classes abstratas, os métodos não podem conter codificação, apenas apresentam assinatura.
- b) () As interfaces contém apenas a assinatura dos métodos
- c) () As subclasses de uma classe abstrata precisam codificar os métodos abstratos das superclasses.
- d) () Uma subclasse ao implementar uma interface pode-se optar implementar apenas alguns dos métodos da interface, os métodos usados no polimorfismo.

Exercício 7) Qual o problema na implementação a seguir?

```
public interface IOTEquipamento {

    public char status;

    public void Ligar()
    { status = 'L'; }

}
```

Exercício 10) Qual o problema na implementação a seguir?

```
public abstract class Robo {

    int x, y;

    public abstract void mover(int vx, int vy)
    {
        x = x + vx;
        y = y + vy;
    }

}
```

Parte 4 – Para implementação de exemplos

Exercício 1) Implemente o cenário a seguir:

O cenário é sobre motores para carros, especificamente os motores (superclasse) e os motores elétricos e a combustão (subclasses).

- Implemente a superclasse MOTOR que possui o atributo potência.
- Implemente as subclasses MOTORELÉTRICO e MOTORCOMBUSTAO. A classe motor elétrico possui a capacidade da bateria (Amperes-Hora), o consumo em amperes por hora e a velocidade média (km/h). O motor a combustão possui os atributos de capacidade do tanque de combustível (litros) e o consumo médio (km/l).
- Implemente o método CalcularAutonomia que indica quantos quilômetros um carro equipado com o motor consegue circular sem precisar ser reabastecido. Faça o método para todas as classes, para a classe Motor o método retorna zero e para as subclasses retornam o cálculo baseado nos atributos do motor.
 - Na implementação do método, imprima uma mensagem que indica qual o método de qual objeto está sendo invocado.
- Crie uma classe principal com um vetor de 5 objetos. Instancie diferentes objetos (da superclasse e subclasse) em cada casa do vetor. Percorra o vetor e calcule as áreas de cada um dos objetos no vetor.

Exercício 1b) Para implementar classes abstratas e interfaces

- Na solução, converta a superclasse para uma classe abstrata e faça as devidas revisões no projeto (por exemplo, nos pontos em que você instancia objetos desta classe).

b) Faça a revisão da superclasse para que o método usado como polimorfismo passe a ser um método abstrato. Houve algum problema ao retirar o código deste método e torná-lo abstrato?

c) Imagine um outro método abstrato para o seu cenário.

- Na superclasse crie a assinatura deste novo método abstrato.

- Ao criar este método houve alguma inconsistência em seu projeto?

- Implemente os métodos abstratos nas classes derivadas para adequar seu projeto.

Atividade 1c) Criando uma interface

- Imagine que este seu objeto é um componente dentro de uma solução de Internet das Coisas (Internet of things). Por exemplo ele pode ser motor em um carro autônomo.

- Estes objetos podem receber comandos de Ligar e Desligar via algum recurso de conectividade.

- Crie uma interface ObjetoIoT que possui os métodos.

b) Implemente a interface nas classes derivadas

- Escolha entre implementar o Ligar e Desligar entre a classe abstrata ou as subclasses e codifique os códigos equivalentes.

c) Na sua classe principal, processe o vetor de objetos para que todos recebam chamadas para os dois métodos.