



MAPA – Material de Avaliação Prática da Aprendizagem

Nome: Cristian Alphawille Ferreira	R.A: 23140092-5
Curso: Análise e Desenvolvimento de Sistemas	
Disciplina: Programação Avançada	

Instruções para Realização da Atividade

1. Revise seu arquivo antes do envio. Certifique-se de que é o arquivo correto, formato correto, se contempla todas as demandas da atividade etc.
2. Após o envio não serão permitidas alterações.
3. Durante a disciplina, procure sanar suas dúvidas pontuais em relação ao conteúdo relacionado à atividade. Porém, não são permitidas correções parciais, ou seja, enviar para que o professor possa fazer uma avaliação prévia e retornar para que o aluno possa ajustar e enviar novamente. Isso não é permitido, pois descaracteriza o processo de avaliação.
4. Ao utilizar quaisquer materiais de pesquisa referencie conforme as normas da ABNT;

Em caso de dúvidas, entre em contato com seu Professor Mediador.

Bons estudos!



AGORA, É COM VOCÊ!

A Programação Orientada a Objetos (POO) é um dos paradigmas mais utilizados no desenvolvimento de software, permitindo a criação de sistemas modulares, reutilizáveis e de fácil manutenção. Em Java, a Programação Orientada a Objetos é baseada em classes, objetos, encapsulamento, herança e polimorfismo, conceitos essenciais para a modelagem de soluções eficientes e escaláveis.

Neste conjunto de atividades, os estudantes praticarão e consolidarão esses fundamentos, avançando progressivamente desde a criação de classes e objetos até a implementação de herança, polimorfismo e interfaces. O objetivo é não apenas compreender a teoria, mas também aplicar os conceitos em situações reais, desenvolvendo habilidades que são fundamentais para o mercado de trabalho.

Referência: OLIVEIRA JUNIOR, E. A.; PEREIRA, R. de L. **Programação Avançada**. Maringá: UniCesumar, 2016.

Classes e Objetos

1 Criando a primeira classe

Crie uma classe chamada Pessoa com os seguintes atributos:

nome (String)

idade (int)

cidade (String)

No método main, instancie um objeto da classe Pessoa e exiba os dados no console.

Métodos e Construtores

2 Classe "Retângulo" com métodos

Crie uma classe Retângulo com os atributos largura e altura e um método:

calcularArea() → Retorna a área do retângulo.

No main, instancie um Retângulo, defina os valores e exiba a área.

3 Conta Bancária (Construtor e Métodos)

Crie a classe ContaBancaria com os atributos:

títular

saldo

Implemente os métodos:

depositar(double valor)

sacar(double valor)

exibirSaldo()

No main, crie um objeto e teste as operações de saque e depósito.

Encapsulamento (Getters e Setters)

4 Classe "Produto" (Encapsulamento)

Crie uma classe Produto com os atributos privados:

nome (String)

preco (double)

Implemente os métodos **getters e setters** para acessar e modificar os atributos.



No main, instancie um objeto Produto, defina os valores e exiba as informações.

Herança e Polimorfismo

5 Criando a classe "Funcionário"

Crie uma classe Funcionário com os atributos:

nome

salario

Crie o método calcularBonus(), que retorna **10% do salário**.

Agora, crie a classe Gerente, que **herda** de Funcionário, e **sobrescreva** calcularBonus() para retornar **20% do salário**.

Teste o polimorfismo no main:

Crie um Funcionário e um Gerente, e exiba o bônus de cada um.

6 Sistema de Veículos (Herança e Polimorfismo)

Crie uma **classe base** chamada Veículo com um método mover().

Agora, crie duas classes-filhas:

Carro → Sobrescreva mover() para exibir "**O carro está se movendo rapidamente**".

Bicicleta → Sobrescreva mover() para exibir "**A bicicleta está se movendo lentamente**".

No main, teste o polimorfismo chamando mover() em objetos Carro e Bicicleta.

Como entregar a atividade:

A atividade deverá ser produzida em um arquivo do tipo TEXTO, conforme TEMPLATE anexado no MATERIAL DA DISCIPLINA, disponibilizado no Studeo, e DEVE ser entregue com a extensão (.PDF). Depois, deve ser anexado no ambiente da Atividade no STUDEO.

Obs.: inserir o print com o código e a execução do projeto.

Dicas para realizar a atividade:

1. Durante as aulas, o professor fornecerá dicas que podem ser utilizadas para a confecção das suas atividades. Assim, é de suma importância participar das aulas ao vivo ou assisti-las posteriormente.
2. Assista às aulas conceituais da disciplina.

Orientações:

Plágios e cópias indevidas serão penalizadas com descontos na nota, podendo chegar a zero.

Não são permitidas correções parciais no decorrer do módulo, pois a interpretação da atividade também faz parte da avaliação.

Atenção ao prazo de entrega da atividade. Sugerimos que envie sua atividade antes do prazo final para evitar transtornos e lentidão nos servidores. Evite o envio de atividade em cima do prazo.

Boa atividade!



Coloque sua resposta a continuação:

Print do código e execução:

Resposta 1 – Classe Pessoa

```
1 public class Pessoa {
2     String nome; 2 usages
3     int idade; 2 usages
4     String cidade; 2 usages
5
6     public static void main(String[] args) {
7         Pessoa pessoa = new Pessoa();
8         pessoa.nome = "Cristian";
9         pessoa.idade = 27;
10        pessoa.cidade = "Toledo";
11
12        System.out.println("Nome: " + pessoa.nome);
13        System.out.println("Idade: " + pessoa.idade);
14        System.out.println("Cidade: " + pessoa.cidade);
15    }
16 }
```

```
C:\Users\cris\.jdk\openjdk-24.0.1\bin\j
Nome: Cristian
Idade: 27
Cidade: Toledo

Process finished with exit code 0
```



Resposta 2 – Classe Retângulo

```
1  ▶ public class Retângulo {  
2      double largura; 2 usages  
3      double altura; 2 usages  
4  
5      public double calcularArea(){ 1 usage  
6          return largura * altura;  
7      }  
8  
9  ▶ public static void main (String[] args){  
10         Retângulo ret = new Retângulo();  
11         ret.largura = 5;  
12         ret.altura = 3;  
13         System.out.println("Área do retângulo: " + ret.calcularArea());  
14     }  
15 }  
16
```

```
C:\Users\cris\jdk\openjdk-24.0.1  
Área do retângulo: 15.0  
  
Process finished with exit code 0
```

Resposta 3 – Conta Bancária

```
1 public class ContaBancaria {
2     String titular; 1 usage
3     double saldo; 4 usages
4
5     public void depositar(double valor){ 1 usage
6         saldo += valor;
7     }
8     public void sacar (double valor){ 1 usage
9         if (valor <= saldo) saldo -= valor;
10        else {
11            System.out.println("Saldo insuficiente.");
12        }
13    }
14    public void exibirSaldo (){ 1 usage
15        System.out.println("Saldo atual: " + saldo);
16    }
17 public static void main (String[] args){
18     ContaBancaria conta = new ContaBancaria();
19     conta.titular = "Cristian";
20
21     conta.depositar( valor: 1000);
22     conta.sacar( valor: 250);
23     conta.exibirSaldo();
24
25 }
26 }
27
28
```



```
C:\Users\crist\.jdk\openjdk-24.0.1\bin\java.exe "-ja
Saldo atual: 750.0





Process finished with exit code 0
```



Resposta 4 – Classe Produto com Encapsulamento

```
1 public class Produto {
2     private String nome; 2 usages
3     private double preco; 2 usages
4
5     public String getNome(){ 1 usage
6         return nome;
7     }
8     public void setNome(String nome){ 1 usage
9         this.nome = nome;
10    }
11    public double getPreco(){ 1 usage
12        return preco;
13    }
14
15    public void setPreco(double preco) { 1 usage
16        this.preco = preco;
17    }
18    public static void main (String[] args){
19        Produto produto = new Produto();
20        produto.setNome("Notebook");
21        produto.setPreco(2500.00);
22
23        System.out.println("Produto: " + produto.getNome());
24        System.out.println("Preco R$: " + produto.getPreco());
25    }
26 }
27
```

Run  Produto 

↑ C:\Users\cris... \openjdk-24.0.1\b
↓ Produto: Notebook
↕ Preço R\$: 2500.0
≡
≡↓ Process finished with exit code 0
📄
>

Resposta 5 – Funcionário e Gerente (Herança e Polimorfismo)

```
class Funcionario { 3 usages 1 inheritor
    double salario; 4 usages

    public double calcularBonus() { 2 usages 1 override
        return salario * 0.10;
    }
}

class Gerente extends Funcionario { 2 usages
    public double calcularBonus() { 2 usages
        return salario * 0.20;
    }
}

public class TesteFuncionario { 1 inheritor
    public static void main(String[] args) {
        Funcionario f = new Funcionario();
        f.nome = "João";
        f.salario = 2000;

        Gerente g = new Gerente();
        g.nome = "Maria";
        g.salario = 5000;
        System.out.println(f.nome + " - Bônus: R$ " + f.calcularBonus());
        System.out.println(g.nome + " - Bônus: R$ " + g.calcularBonus());
    }
}
```

```
Run TesteFuncionario (1) x
C:\Users\crist\.jdk\openjdk-24.0.1\bin
João - Bônus: R$ 200.0
Maria - Bônus: R$ 1000.0
Process finished with exit code 0
```




Resposta 6 – Sistema de Veículos (Herança e Polimorfismo)

```
class Veiculo { 4 usages 2 inheritors
    public void mover() { 2 usages 2 overrides
        System.out.println("0 veiculo está se movendo.");
    }
}

class Carro extends Veiculo { 1 usage
    @Override 2 usages
    public void mover() {
        System.out.println("0 carro está se movendo rapidamente.");
    }
}

class Bicicleta extends Veiculo { 1 usage
    @Override 2 usages
    public void mover() {
        System.out.println("A bicicleta está se movendo lentamente.");
    }
}

public class TesteVeiculos {
    public static void main(String[] args) {
        Veiculo carro = new Carro();
        Veiculo bicicleta = new Bicicleta();

        carro.mover();
        bicicleta.mover();
    }
}
```

```
C:\Users\crist\.jdk\openjdk-24.0.1\bin\j
0 carro está se movendo rapidamente.
A bicicleta está se movendo lentamente.

Process finished with exit code 0
```

**Importante:**

- Certifique-se de que todos os prints estejam legíveis e mostrem a execução correta do projeto.
- O arquivo final deve ser exportado em formato **.PDF**, conforme orientações da disciplina.