



MAPA – Material de Avaliação Prática da Aprendizagem

Nome: Francisco Willians Da Vera Cruz Silva	R.A 23102201-5
Curso: Superior De Tecnologia Em 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.

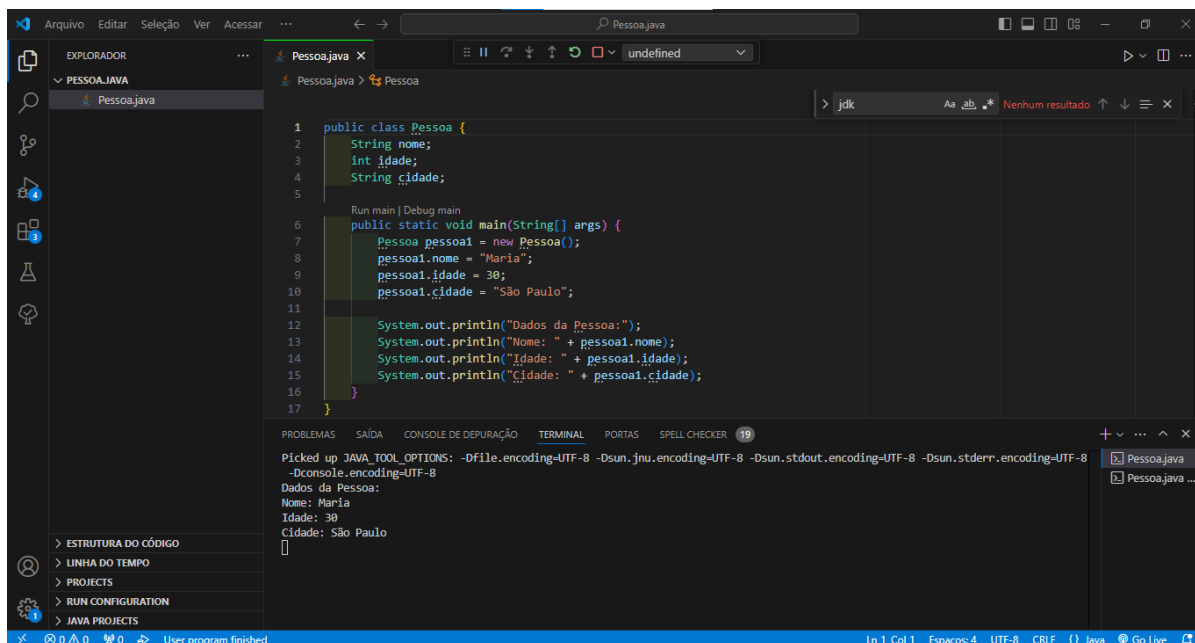
Boa atividade!



Coloque sua resposta a continuação:

Print do código e execução:

Resposta 1 – Classe Pessoa



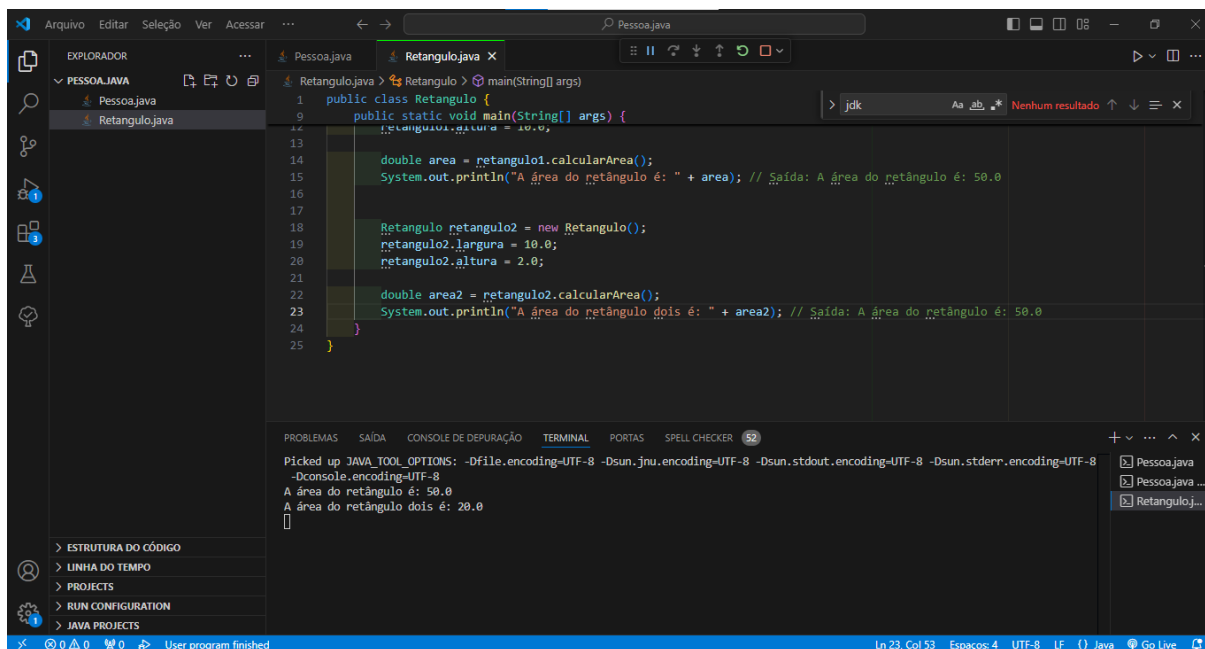
```
1 public class Pessoa {
2     String nome;
3     int idade;
4     String cidade;
5
6     Run main | Debug main
7     public static void main(String[] args) {
8         Pessoa pessoa1 = new Pessoa();
9         pessoa1.nome = "Maria";
10        pessoa1.idade = 30;
11        pessoa1.cidade = "São Paulo";
12
13        System.out.println("Dados da Pessoa:");
14        System.out.println("Nome: " + pessoa1.nome);
15        System.out.println("Idade: " + pessoa1.idade);
16        System.out.println("Cidade: " + pessoa1.cidade);
17    }
18 }
```

PROBLEMAS SAÍDA CONSOLE DE DEPUÇÃO TERMINAL PORTAS SPELL CHECKER 19

Picked up JAVA_TOOL_OPTIONS: -Dfile.encoding=UTF-8 -Dsun.jnu.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8
-Dconsole.encoding=UTF-8
Dados da Pessoa:
Nome: Maria
Idade: 30
Cidade: São Paulo

Ln 1, Col 1 Espaços: 4 UTF-8 CRLF {} Java Go Live

Resposta 2 – Classe Retângulo



```
1 public class Retangulo {
2     public static void main(String[] args) {
3         Retangulo r1 = new Retangulo();
4         r1.largura = 10.0;
5         r1.altura = 5.0;
6
7         double area = r1.calcularArea();
8         System.out.println("A área do retângulo é: " + area); // Saída: A área do retângulo é: 50.0
9
10        Retangulo r2 = new Retangulo();
11        r2.largura = 10.0;
12        r2.altura = 2.0;
13
14        double area2 = r2.calcularArea();
15        System.out.println("A área do retângulo dois é: " + area2); // Saída: A área do retângulo é: 20.0
16    }
17 }
```

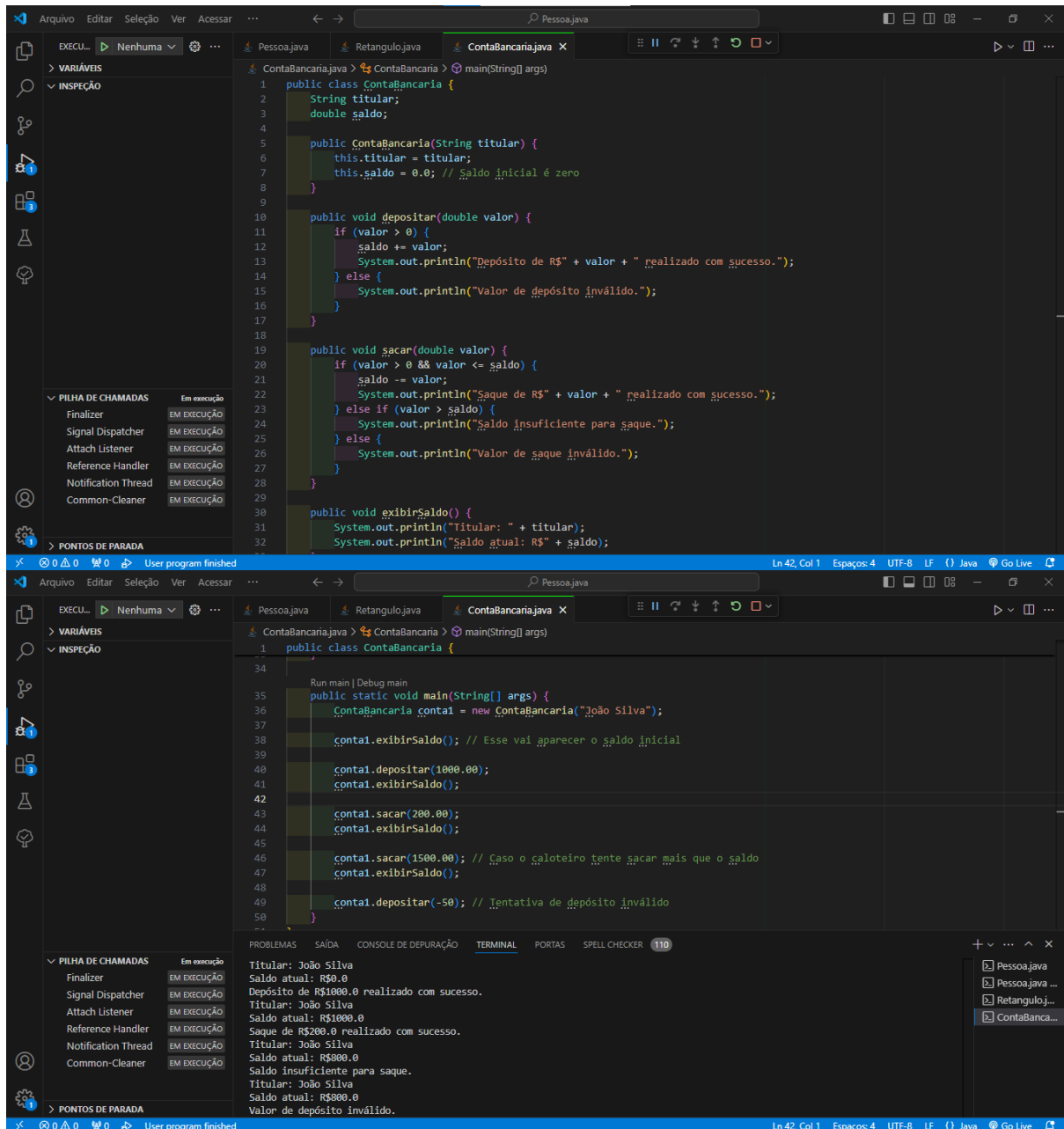
PROBLEMAS SAÍDA CONSOLE DE DEPUÇÃO TERMINAL PORTAS SPELL CHECKER 52

Picked up JAVA_TOOL_OPTIONS: -Dfile.encoding=UTF-8 -Dsun.jnu.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8
-Dconsole.encoding=UTF-8
A área do retângulo é: 50.0
A área do retângulo dois é: 20.0

Ln 23, Col 53 Espaços: 4 UTF-8 LF {} Java Go Live



Resposta 3 – Conta Bancária



The screenshot displays an IDE with the following components:

- Top Panel:** Shows the file explorer with 'ContaBancaria.java' selected.
- Editor:** Contains the Java code for the 'ContaBancaria' class.


```

1 public class ContaBancaria {
2     String titular;
3     double saldo;
4
5     public ContaBancaria(String titular) {
6         this.titular = titular;
7         this.saldo = 0.0; // Saldo inicial é zero
8     }
9
10    public void depositar(double valor) {
11        if (valor > 0) {
12            saldo += valor;
13            System.out.println("Depósito de R$ " + valor + " realizado com sucesso.");
14        } else {
15            System.out.println("Valor de depósito inválido.");
16        }
17    }
18
19    public void sacar(double valor) {
20        if (valor > 0 && valor <= saldo) {
21            saldo -= valor;
22            System.out.println("Saque de R$ " + valor + " realizado com sucesso.");
23        } else if (valor > saldo) {
24            System.out.println("Saldo insuficiente para saque.");
25        } else {
26            System.out.println("Valor de saque inválido.");
27        }
28    }
29
30    public void exibirSaldo() {
31        System.out.println("Titular: " + titular);
32        System.out.println("Saldo atual: R$ " + saldo);
33    }
34
35    public static void main(String[] args) {
36        ContaBancaria conta = new ContaBancaria("João Silva");
37        conta.exibirSaldo(); // Esse vai aparecer o saldo inicial
38        conta.depositar(1000.00);
39        conta.exibirSaldo();
40        conta.sacar(200.00);
41        conta.exibirSaldo();
42        conta.sacar(1500.00); // Caso o galoteiro tente sacar mais que o saldo
43        conta.exibirSaldo();
44        conta.depositar(-50); // Tentativa de depósito inválido
45    }
46
47 }
      
```
- Bottom Panel:** Shows the 'TERMINAL' output of the program execution.


```

Titular: João Silva
Saldo atual: R$0.0
Depósito de R$1000.0 realizado com sucesso.
Titular: João Silva
Saldo atual: R$1000.0
Saque de R$200.0 realizado com sucesso.
Titular: João Silva
Saldo atual: R$800.0
Saldo insuficiente para saque.
Titular: João Silva
Saldo atual: R$800.0
Valor de depósito inválido.
      
```

Resposta 4 – Classe Produto com Encapsulamento

```

1 public class Produto {
2     private String nome;
3     private double preco;
4
5
6     public String getNome() {
7         return nome;
8     }
9
10
11    public void setNome(String nome) {
12        this.nome = nome;
13    }
14
15
16    public double getPreco() {
17        return preco;
18    }
19
20

```

Informações do Produto:
Nome: Notebook Gamer
Preço: R\$5500.99
Preço inválido. O preço não pode ser negativo.
Preço após tentativa de valor inválido: R\$5500.99

```

18 }
19
20
21 public void setPreco(double preco) {
22     if (preco >= 0) {
23         this.preco = preco;
24     } else {
25         System.out.println("Preço inválido. O preço não pode ser negativo.");
26     }
27 }
28
29 Run main | Debug main
30 public static void main(String[] args) {
31     Produto produto1 = new Produto();
32
33     produto1.setNome("Notebook Gamer");
34     produto1.setPreco(5500.99);
35
36     // aqui va exibir as informações usando os getters
37     System.out.println("Informações do Produto:");
38     System.out.println("Nome: " + produto1.getNome());
39     System.out.println("Preço: R$" + produto1.getPreco());
40
41     // tentativa de definir um preço inválido
42     produto1.setPreco(-100.0);
43     System.out.println("Preço após tentativa de valor inválido: R$" + produto1.getPreco());
44 }
45 }

```

Informações do Produto:
Nome: Notebook Gamer
Preço: R\$5500.99
Preço inválido. O preço não pode ser negativo.
Preço após tentativa de valor inválido: R\$5500.99

Informações do Produto:
Nome: Notebook Gamer
Preço: R\$5500.99
Preço inválido. O preço não pode ser negativo.
Preço após tentativa de valor inválido: R\$5500.99

Resposta 5 – Funcionário e Gerente (Herança e Polimorfismo)

Funcionario.java

```

1 class Funcionario {
2     String nome;
3     double salario;
4
5     public Funcionario(String nome, double salario) {
6         this.nome = nome;
7         this.salario = salario;
8     }
9     public double calcularBonus() {
10        return this.salario * 0.10; // 10%
11    }
12    public String getNome() {
13        return nome;
14    }
15    public double getSalario() {
16        return salario;
17    }
18 }
19 class Gerente extends Funcionario {
20
21     public Gerente(String nome, double salario) {
22         super(nome, salario);
23     }
24
25     @Override
26     public double calcularBonus() {
27         return this.salario * 0.20; // 20%
28     }
29 }

```

PolimorfismoFuncionario.java

```

1 public class PolimorfismoFuncionario {
2     public static void main(String[] args) {
3         Funcionario funcionarioComum = new Funcionario(nome:"Carlos Alberto", salario:3000.00);
4         Gerente gerenteLoja = new Gerente(nome:"Ana Paula", salario:7000.00);
5
6         System.out.println("Funcionário: " + funcionarioComum.getNome());
7         System.out.println("Salário: R$" + funcionarioComum.getSalario());
8         System.out.println("Bônus: R$" + funcionarioComum.calcularBonus());
9
10        System.out.println("\nGerente: " + gerenteLoja.getNome());
11        System.out.println("Salário: R$" + gerenteLoja.getSalario());
12        System.out.println("Bônus: R$" + gerenteLoja.calcularBonus());
13
14        System.out.println("\n--- Teste de Polimorfismo ---");
15        Funcionario f1 = funcionarioComum;
16        Funcionario f2 = gerenteLoja;
17
18        System.out.println("Bônus (f1 - Funcionário): R$" + f1.calcularBonus());
19        System.out.println("Bônus (f2 - Gerente): R$" + f2.calcularBonus());
20    }
21 }

```

Terminal Output:

```

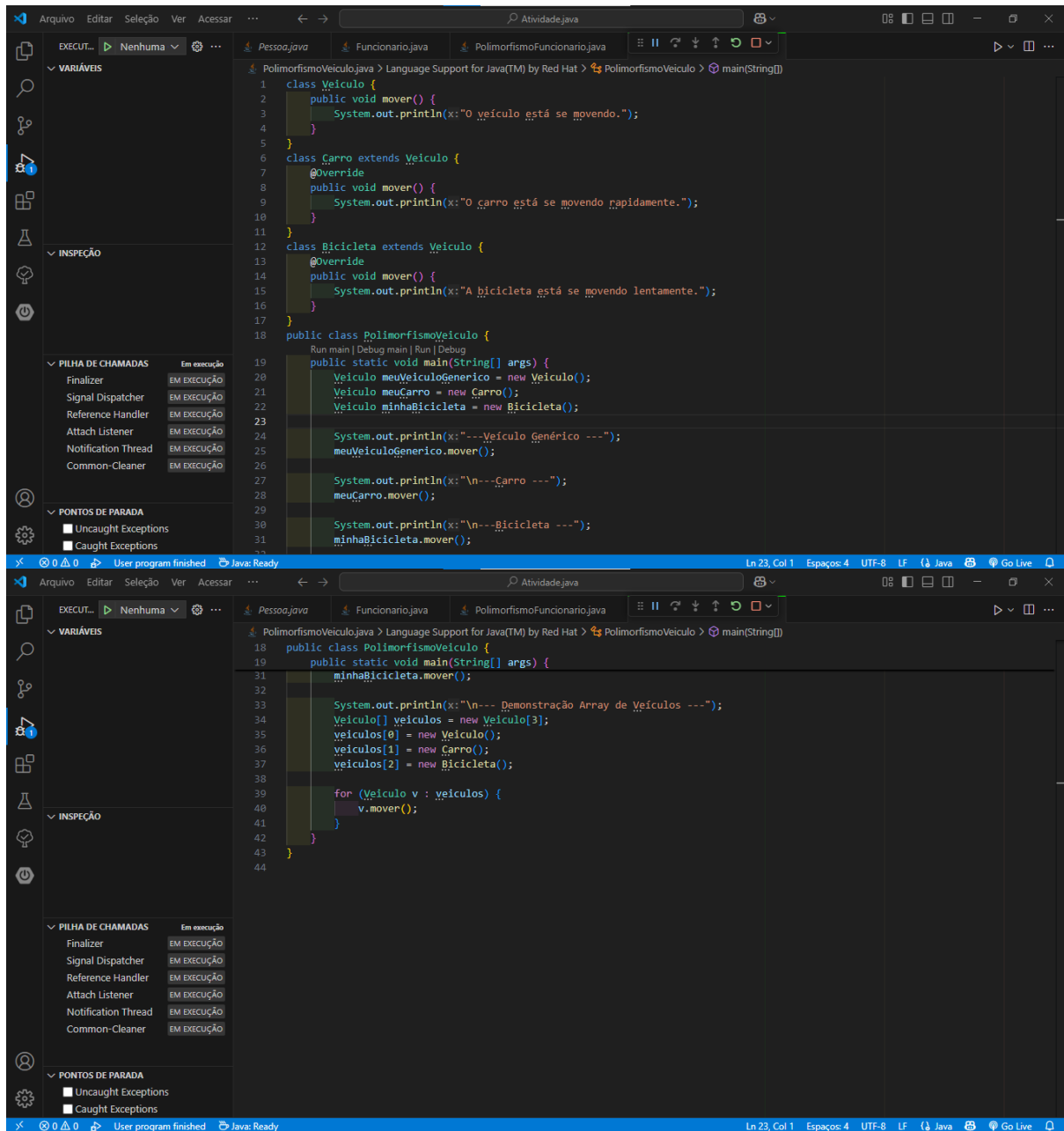
Picked up JAVA_TOOL_OPTIONS: -Dfile.encoding=UTF-8 -Dsun.jnu.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8
-Dconsole.encoding=UTF-8
Funcionário: Carlos Alberto
Salário: R$3000.0
Bônus: R$300.0

Gerente: Ana Paula
Salário: R$7000.0
Bônus: R$1400.0

--- Teste de Polimorfismo ---
Bônus (f1 - Funcionário): R$300.0
Bônus (f2 - Gerente): R$1400.0

```

Resposta 6 – Sistema de Veículos (Herança e Polimorfismo)



The image shows two screenshots of an IDE (IntelliJ IDEA) displaying Java code for a vehicle system. The top screenshot shows the initial code structure, and the bottom screenshot shows the code after adding a loop to process an array of vehicles.

Top Screenshot Code:

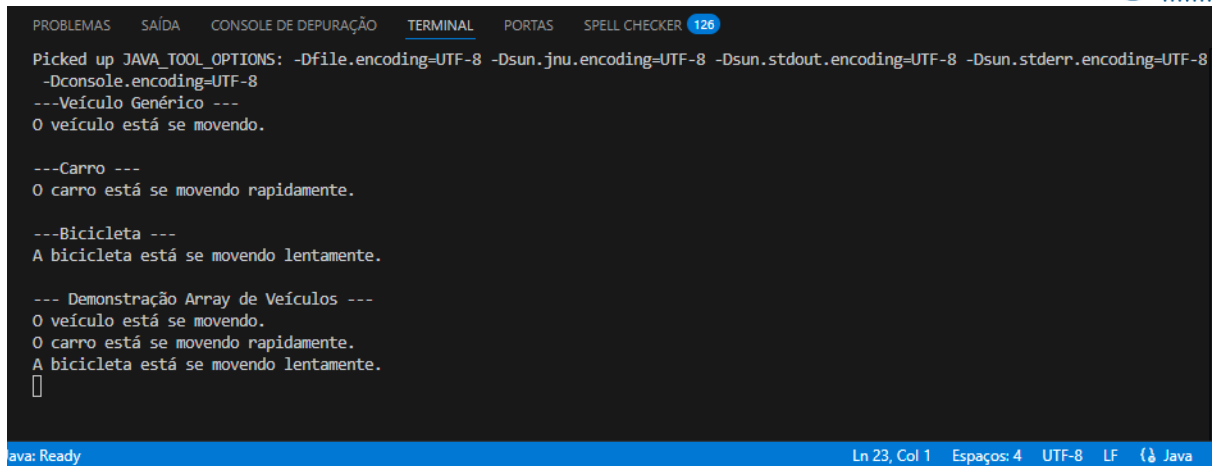
```

1 class Veiculo {
2     public void mover() {
3         System.out.println("O veículo está se movendo.");
4     }
5 }
6 class Carro extends Veiculo {
7     @Override
8     public void mover() {
9         System.out.println("O carro está se movendo rapidamente.");
10    }
11 }
12 class Bicicleta extends Veiculo {
13     @Override
14     public void mover() {
15         System.out.println("A bicicleta está se movendo lentamente.");
16    }
17 }
18 public class PolimorfismoVeiculo {
19     Run main | Debug main | Run | Debug
20     public static void main(String[] args) {
21         Veiculo meuVeiculoGenerico = new Veiculo();
22         Veiculo meuCarro = new Carro();
23         Veiculo minhaBicicleta = new Bicicleta();
24
25         System.out.println("----Veículo Genérico ----");
26         meuVeiculoGenerico.mover();
27
28         System.out.println("\n---Carro ---");
29         meuCarro.mover();
30
31         System.out.println("\n---Bicicleta ---");
32         minhaBicicleta.mover();
33     }
34 }
    
```

Bottom Screenshot Code:

```

18 public class PolimorfismoVeiculo {
19     public static void main(String[] args) {
20         minhaBicicleta.mover();
21
22         System.out.println("\n--- Demonstração Array de Veículos ---");
23         Veiculo[] veiculos = new Veiculo[3];
24         veiculos[0] = new Veiculo();
25         veiculos[1] = new Carro();
26         veiculos[2] = new Bicicleta();
27
28         for (Veiculo v : veiculos) {
29             v.mover();
30         }
31     }
32 }
    
```

```
PROBLEMAS  SAÍDA  CONSOLE DE DEPURAÇÃO  TERMINAL  PORTAS  SPELL CHECKER  126

Picked up JAVA_TOOL_OPTIONS: -Dfile.encoding=UTF-8 -Dsun.jnu.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8
-Dconsole.encoding=UTF-8
---Veículo Genérico ---
O veículo está se movendo.

---Carro ---
O carro está se movendo rapidamente.

---Bicicleta ---
A bicicleta está se movendo lentamente.

--- Demonstração Array de Veículos ---
O veículo está se movendo.
O carro está se movendo rapidamente.
A bicicleta está se movendo lentamente.
█

ava: Ready  Ln 23, Col 1  Espaços: 4  UTF-8  LF  Java
```

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.