

Lista de Exercícios de Revisão – Algoritmos e Programação em Java

INSTRUÇÕES GERAIS:

1. Todos os códigos devem ser escritos em Java e conter comentários explicando o funcionamento de cada função, variável e bloco lógico.

2. Os exercícios devem ser feitos individualmente e entregues feitos à mão.

1. Crie uma classe `Pessoa` com nome e idade. Instancie dois objetos e exiba seus dados.
2. Crie uma classe `Produto` com nome, preço e quantidade. Faça um método que calcule o valor total em estoque.
3. Implemente uma classe `Carro` com marca, modelo e ano. Exiba os dados no console.
4. Crie uma classe `Livro` com título, autor e número de páginas. Adicione um método que mostre uma mensagem personalizada.
5. Crie uma classe `ContaBancaria` com atributos titular e saldo. Faça métodos para depositar e sacar.
6. Crie uma classe `Funcionario` com nome e salário. Adicione um método que calcule o bônus de 10%.
7. Crie uma classe `Aluno` com notas e um método que calcule a média.
8. Crie uma classe `Retangulo` com base e altura e um método que calcule a área.
9. Crie uma classe `Triangulo` e verifique se os lados formam um triângulo válido.
10. Crie uma classe `Temperatura` que converta Celsius para Fahrenheit.
11. Adicione `get` e `set` à classe `ContaBancaria` e restrinja saques com valor maior que o saldo.
12. Crie uma classe `Computador` com atributos privados e métodos de acesso.
13. Implemente uma classe `Pessoa` que só permita idades entre 0 e 120.
14. Crie uma classe `Carro` que só permita ligar se tiver combustível.
15. Crie uma classe `Elevador` que controle o andar atual, o máximo e o mínimo permitido.
16. Adicione validações à classe `Aluno` para garantir notas entre 0 e 10.
17. Crie uma classe `Lampada` com métodos `ligar` e `desligar` e uma variável booleana de estado.
18. Crie uma classe `Caixa` que calcule o volume e tenha atributos privados.
19. Implemente uma classe `Musica` com nome e duração e mostre o tempo total de várias músicas.
20. Crie uma classe `Pedido` que some o valor total de vários produtos.

21. Crie um construtor em `Aluno` que receba nome e matrícula.
22. Crie um construtor em `Produto` que receba nome, preço e quantidade.
23. Faça sobrecarga de construtores em `Livro` (um com todos os dados e outro apenas com o título).
24. Implemente uma sobrecarga de método `depositar` em `ContaBancaria` (um com `double` e outro com `int`).
25. Crie uma classe `Veiculo` com construtor padrão e outro parametrizado.
26. Crie uma classe `Cachorro` com dois construtores — um com nome, outro com nome e idade.
27. Implemente sobrecarga do método `calcularMedia` em `Aluno` (com 2 e 3 notas).
28. Crie uma classe `Pessoa` que possa ser inicializada com nome e CPF, ou apenas nome.
29. Crie um método `mostrarInfo` sobrecarregado para exibir dados básicos ou completos.
31. Crie uma superclasse `Pessoa` e duas subclasses `Professor` e `Aluno`.
32. Faça com que `Professor` tenha atributo `disciplina` e `Aluno` tenha `curso`.
33. Crie uma classe `Animal` e subclasses `Cachorro` e `Gato` que sobrescrevam um método `falar()`.
34. Crie uma classe `Funcionario` e subclasses `Gerente` e `Estagiario`.
35. Crie uma classe `Veiculo` e subclasses `Carro` e `Moto` que calculem o consumo.
36. Crie uma classe `Forma` e subclasses `Quadrado` e `Circulo` com métodos `calcularArea`.
37. Crie uma classe `Pessoa` e sobrescreva o método `toString()` nas subclasses.
38. Implemente uma hierarquia `Conta` → `ContaCorrente` e `ContaPoupanca` com taxas diferentes.
39. Crie uma hierarquia `Funcionario` → `Programador` e `Designer` com salários diferentes.
40. Crie um exemplo de polimorfismo exibindo o método `mostrarDados()` para diferentes objetos.
41. Crie uma classe `Pagamento` com método `processar()` e subclasses `Pix`, `Cartao`, `Boleto`.
42. Faça com que cada classe do exercício anterior imprima uma mensagem diferente.
43. Crie uma classe `Mensagem` com método `enviar()` e subclasses `Email` e `SMS`.
44. Crie uma classe `Funcionario` com método `calcularBonus()` sobrescrito em `Gerente` e `Vendedor`.
45. Crie uma classe `Veiculo` e sobrescreva o método `acelerar()` em `Carro` e `Moto`.
46. Crie uma classe `Animal` com método `emitirSom()` e subclasses que o implementem de formas diferentes.
47. Crie um método polimórfico que receba qualquer tipo de `Animal` e invoque `emitirSom()`.
48. Crie uma lista de `Funcionario` e calcule o total de bônus usando polimorfismo.
49. Crie uma classe `Imposto` e subclasses `PessoaFisica` e `PessoaJuridica` com regras diferentes.

50. Crie um programa que demonstre o uso de polimorfismo em tempo de execução.
51. Crie um vetor de 5 notas e calcule a média.
52. Crie um vetor de nomes e exiba-os em ordem inversa.
53. Crie uma matriz 3x3 e exiba a diagonal principal.
54. Crie uma matriz 4x4 e calcule a soma dos elementos.
55. Crie um vetor de 10 inteiros e exiba o maior e o menor.
56. Crie um programa que leia 5 notas e informe a quantidade de alunos aprovados.
57. Crie uma matriz de 3 alunos x 3 notas e calcule a média de cada aluno.
58. Crie uma matriz de temperaturas semanais e exiba a maior temperatura.
59. Crie um vetor com 5 nomes e verifique se um nome específico está presente.
60. Crie um programa que leia uma matriz 2x2 e exiba a soma das diagonais.
61. Explique o que é uma classe e um objeto.
62. Qual é a diferença entre variável de instância e variável local?
63. O que significa encapsulamento em POO?
64. Por que usamos métodos `get` e `set`?
65. Qual é a função do construtor em uma classe?
66. Explique o conceito de sobrecarga de método.
67. Qual a diferença entre `this` e o nome do atributo?
68. O que acontece se um construtor não for declarado?
69. Como o Java trata atributos privados em classes?
70. Por que é importante comentar o código?
71. O que é herança em Java?
72. O que significa sobrescrita de método?
73. Qual a diferença entre herança e composição?
74. O que é uma superclasse e uma subclasse?
75. Como o polimorfismo melhora a flexibilidade do código?
76. Quando o `super()` é utilizado?
77. O que acontece se uma subclasse não sobrescrever um método da superclasse?
78. Explique um exemplo prático de polimorfismo.
79. Como a herança favorece o reuso de código?
80. Cite um risco de usar herança de forma inadequada.
81. Qual a diferença entre vetor e matriz?
82. Como acessar o último elemento de um vetor?
83. O que acontece se tentarmos acessar uma posição inexistente?
84. Qual a vantagem de usar laços `for` em vetores?
85. Explique o conceito de índice em uma estrutura de repetição.
86. O que é uma matriz tridimensional?
87. Como percorrer uma matriz bidimensional?
88. Qual a utilidade de arrays em aplicações reais?
89. O que significa inicializar um vetor?
90. Como combinar vetores e objetos?

91. Por que separar código em classes diferentes é importante?
92. O que significa coesão e acoplamento?
93. Por que usar comentários descritivos em métodos?
94. Quando se deve usar herança múltipla? (Explique por que Java não permite diretamente.)
95. Qual a diferença entre `static` e não estático?
96. Por que é importante testar métodos individualmente?
97. Como os conceitos de POO aparecem em sistemas do mundo real?
98. O que é uma instância de classe?
99. Por que devemos evitar variáveis públicas?
100. Qual o papel do método `main` em um programa Java orientado a objetos?

Se for chorar, manda áudio!
Disse o filósofo