Lista de Exercícios - Paradigma Orientado a Objetos

1. Classes e Objetos Básicos

Implemente uma classe `Pessoa` com atributos para nome, idade e um método para exibir as informações da pessoa.

2. Encapsulamento

Implemente uma classe `ContaBancaria` com atributos privados `saldo` e métodos públicos `depositar`, `sacar` e `consultarSaldo`.

3. Construtores

Implemente uma classe `Produto` que possui um construtor que aceita nome, preço e quantidade em estoque. Crie instâncias de `Produto` usando diferentes valores.

4. Métodos e Sobrecarga

Implemente uma classe `Calculadora` com métodos para adicionar, subtrair, multiplicar e dividir. Sobrecargue os métodos para operar com diferentes números de parâmetros.

5. Herança Simples

Crie uma classe `Veiculo` com atributos básicos como marca e modelo. Em seguida, crie uma classe `Carro` que herda de `Veiculo` e adicione atributos e métodos específicos para carros.

6. Polimorfismo

Implemente uma classe `Animal` com um método `emitirSom()`. Crie subclasses `Cachorro` e `Gato` que sobrepõem `emitirSom()` para retornar sons específicos.

7. Composição

Crie uma classe `Motor` e uma classe `Carro`. A classe `Carro` deve ter um atributo do tipo `Motor` e métodos para interagir com o motor.

8. Associação e Agregação

Implemente uma classe `Escola` que contém uma lista de objetos da classe `Aluno`. A classe `Aluno` deve ter atributos como nome, matrícula e nota.

9. Herança Múltipla (através de interfaces)

Implemente interfaces `Movimentavel` e `Desenhavel`. Crie uma classe `Personagem` que implementa ambas as interfaces, com métodos para mover e desenhar o personagem.

10. Classes Abstratas

Implemente uma classe abstrata `FormaGeometrica` com um método abstrato `calcularArea()`. Crie subclasses `Quadrado` e `Circulo` que implementam o método `calcularArea()`.

11. Métodos Estáticos

Crie uma classe `MatematicaUtil` com métodos estáticos para calcular o quadrado, cubo e raiz quadrada de um número.

12. Enums

Implemente uma enumeração `DiasDaSemana` e utilize-a em uma classe `Agenda` para marcar compromissos em diferentes dias.

13. Tratamento de Exceções

Crie uma classe `Divisao` que tenha um método para dividir dois números inteiros. Lance uma exceção personalizada caso ocorra divisão por zero.

14. Coleções (ArrayList ou equivalente)

Implemente uma classe `Biblioteca` que contém uma lista de objetos `Livro`. Adicione métodos para adicionar, remover e listar livros.

15. Iteradores e Loops

Expanda o exercício anterior para permitir a iteração sobre a lista de livros utilizando um `Iterator` (ou equivalente na linguagem escolhida).

16. Generics

Crie uma classe genérica `Caixa<T>` que pode armazenar e retornar qualquer tipo de objeto. Implemente métodos para adicionar e remover itens da caixa.

19. Desenvolvimento de um Sistema de Gestão de Funcionários

Desenvolva um sistema básico que permite gerenciar funcionários, incluindo funcionalidades para adicionar, remover, atualizar e listar informações dos funcionários. Não utilize banco de dados; armazene os dados em memória.