**Os exercícios abaixo são um teste, deve-se fazer o código e disponibilizá-lo em .zip ou código de repositório git, e devolver para o e-mail** [**dannel\_kayke@hotmail.com**](mailto:dannel_kayke@hotmail.com) **até as 12h do dia 20/11/2023.**

**Criação de Classe e Instância:**

* Crie uma classe Carro com propriedades como modelo, ano e cor.
* Instancie dois objetos dessa classe com diferentes valores.
* Imprima no console os detalhes de cada carro.

**Herança e Polimorfismo:**

* Crie uma classe base chamada Animal com métodos como emitirSom e mover.
* Derive duas classes, Cachorro e Pássaro, que herdam da classe Animal.
* Sobrescreva o método emitirSom em ambas as classes derivadas.
* Crie instâncias de Cachorro e Pássaro e chame seus métodos.

**Encapsulamento e Métodos Estáticos:**

* Crie uma classe Calculadora com métodos de operações matemáticas (soma, subtração, multiplicação, divisão).
* Encapsule as operações matemáticas, permitindo apenas o acesso através dos métodos da classe.
* Adicione um método estático que retorna o valor absoluto de um número.

**Interfaces e Implementação:**

* Crie uma interface FormaGeometrica com métodos como calcularArea e calcularPerimetro.
* Implemente a interface em classes como Quadrado e Círculo.
* Instancie objetos dessas classes e chame seus métodos.

**Composição de Objetos:**

* Crie uma classe Motor com métodos como ligar e desligar.
* Em seguida, crie uma classe Carro que possui uma instância de Motor.
* Implemente métodos em Carro que delegam chamadas aos métodos correspondentes em Motor.
* Teste a funcionalidade ligando e desligando o carro.

**Tratamento de Exceções:**

* Crie uma classe ContaBancaria com propriedades como saldo e métodos como sacar e depositar.
* Implemente uma verificação para garantir que o saldo não fique negativo após um saque.
* Utilize exceções para lidar com situações em que o saque não pode ser realizado devido a saldo insuficiente.
* Teste a classe com casos que resultem em exceções.

**Padrões de Projeto: Singleton:**

* Implemente uma classe ConfiguracaoApp usando o padrão Singleton, garantindo que exista apenas uma instância dessa classe.
* Adicione propriedades de configuração à classe e métodos para acessá-las.
* Tente criar várias instâncias da classe e verifique se todas se referem à mesma instância.

**PROJETO INDIVIDUAL**

**Sistema de Gerenciamento de Biblioteca**

Considere o desenvolvimento de um sistema de gerenciamento de biblioteca em TypeScript. Este sistema deve permitir a gestão de livros, autores e usuários. Os requisitos básicos são:

**Livro:**

* Um livro possui atributos como título, autor, anoPublicacao, e genero.
* Implemente métodos para emprestar e devolver livros.

**Autor:**

* Um autor possui atributos como nome, dataNascimento e nacionalidade.
* Implemente métodos para adicionar e remover livros associados ao autor.

**Usuário:**

* Um usuário possui atributos como nome, email e livrosEmprestados.
* Implemente métodos para emprestar e devolver livros.

**Biblioteca:**

* A biblioteca é responsável por manter registros de livros, autores e usuários.
* Implemente métodos para adicionar e remover livros, autores e usuários.
* Forneça métodos para buscar livros por autor, listar livros emprestados, etc.

**Relatórios:**

* Crie um mecanismo para gerar relatórios, como listar todos os livros emprestados, livros disponíveis, etc.

**Testes:**

* Desenvolva testes unitários para garantir a integridade do sistema.
* Considere cenários como tentativa de empréstimo de livro indisponível, adição de autor sem livros associados, etc.

**Requisitos Técnicos:**

* Utilize classes e interfaces para modelar livros, autores e usuários.
* Implemente métodos construtores, getters, setters e métodos específicos para cada classe.
* Utilize herança e composição conforme apropriado.
* Considere a utilização de decorators para adicionar funcionalidades específicas, como logging de operações.

**Observações:**

Não existe código “mais correto”, existe abstração! Crie utilizando as melhores práticas de Orientações a Objetos**.**

**Não se preocupe caso falte implementação de alguma funcionalidade, faça o melhor possível, o importante é aprender e demonstrar o que entendeu!**