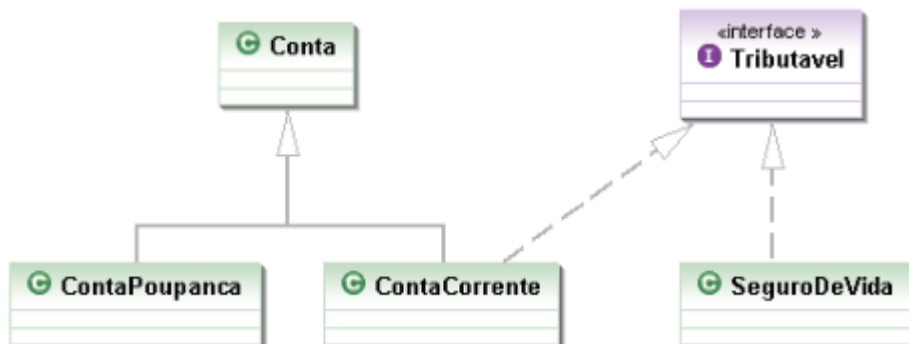


LISTA DE EXERCÍCIOS

Disciplina: PROGRAMAÇÃO ESTRUTURADA E ORIENTADA A OBJETOS

1. Faça o que pede a questão.
 - a) Crie um projeto interfaces e crie a interface `AreaCalculavel` com o método `calculaArea()` - sem parâmetros e que retorna um **double**.
 - b) Queremos criar algumas classes que implementam `AreaCalculavel`:
 - I. Quadrado: possui um atributo `lado`.
 - II. Retângulo: possui os atributos `base` e `altura`.
 - III. Circulo: possui o atributo `raio`.
 - c) Crie uma classe de Teste. No método `main` crie um vetor de 5 posições que contém alguns objetos do tipo `AreaCalculavel`. Logo após, percorra esse vetor imprimindo a área de cada objeto.
2. Nosso banco precisa tributar dinheiro de alguns bens que nossos clientes possuem. Para isso, vamos criar um sistema para isso.



- a) Crie uma interface `Tributavel` que possui o método `calculaTributos()`, que retorna um **double**.
- b) Alguns bens são tributáveis e outros não, `ContaPoupanca` não é tributável, já para `ContaCorrente` você precisa pagar 1% da conta e o `SeguroDeVida` tem uma taxa fixa de 42 reais.
- c) As classes `ContaCorrente` e `ContaPoupanca` herdam de uma classe `Conta`. Essa classe `Conta` possui um saldo e os métodos `sacar(double)`, `depositar(double)` e `obterSaldo()` que retorna o saldo da conta.
- d) Vamos criar uma classe `TestaTributavel` com um método `main` para testar o nosso exemplo.

3. Crie um `GerenciadorDeImpostoDeRenda`, que recebe todos os tributáveis de uma pessoa e soma seus valores e inclua nele um método para devolver seu total. Essa classe deve ter um atributo para calcular a soma total dos tributos e um método `adicionar(Tributavel)` que recebe como parâmetro um `Tributavel` e soma os tributos dele ao total. Crie um main para instanciar diversas classes que implementam `Tributavel` e passar como argumento para um `GerenciadorDeImpostoDeRenda`. Repare que você não pode passar qualquer tipo de conta para o método `adiciona`, apenas as que implementam `Tributavel`.