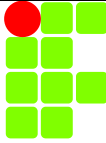


|  |  |
|--|--|
|  <p>INSTITUTO FEDERAL DE<br/>EDUCAÇÃO, CIÊNCIA E TECNOLOGIA<br/>PIAUI</p> | <p><b>INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E<br/>TECNOLOGIA DO PIAUÍ</b></p> <p><b>Curso: Análise e Desenvolvimento de Sistemas</b></p> <p><b>Disciplina: Programação Orientada a Objetos</b></p> <p><b>Professor: Ely</b></p> |
|--|--|

## Exercício 09

### Classes Abstratas

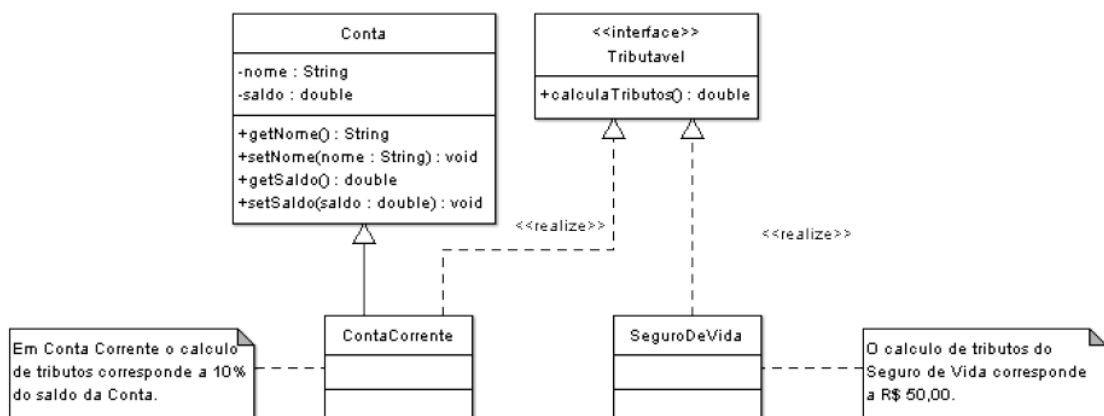
- Podemos instanciar classes abstratas? Justifique.
- Explique o que é necessário para que a compilação da ClasseConcreta ocorra sem erros:

|   |  |
|---|--|
| <pre>abstract class ClasseAbstrata {     abstract imprimaAlgo(): void ; }</pre> | <pre>class ClasseConcreta extends     ClasseAbstrata { }</pre> |
|---|--|

- Se uma classe que herda de uma abstrata e não implementa os seus métodos, o que ocorre?
- Imagine que você deve modelar várias figuras geométricas em TypeScript e que cada uma tem sua forma específica de calcular área e perímetro. Proponha e implemente uma hierarquia de classes usando uma classe abstrata chamada FiguraGeometrica e outras concretas: Quadrado, Triangulo, etc.
- Não podemos aplicar o operador new em FiguraGeometrica, mas por que então podemos dar new em FiguraGeometrica[10], por exemplo?
- Implemente as classes Funcionario, Gerente e Diretor conforme o diagrama exposto em sala:
  - A classe funcionário deve ser abstrata e o método getBonificacao() abstrato;
  - Na classe gerente o método bonificação deve retornar 40% do salário;
  - Em Diretor a bonificação deve ser 60% do salário.
  - Por fim, na classe presidente o método deve retornar 100% do salário + R\$ 1.000,00.

## Interfaces

7. Refaça a questão 04 do exercício usando interfaces com os métodos propostos em vez de herança. Crie também uma classe de teste que instancie e teste diferentes formas geométricas.
8. Crie uma interface chamada IComparavel com um método chamado comparar que receba uma forma geométrica como parâmetro e retorna um inteiro como resultado. Implemente em cada uma das classes do exemplo anterior a interface retornando -1, 0 e 1 caso a área da forma seja menor, igual ou maior que a passada via parâmetro.
9. Crie uma classe para testar os exemplos anteriores. Instancie várias formas diferentes. Pegue duas formas chame em uma delas o método comparar passando a outra como parâmetro e exiba o resultado. Repita para outras formas.
10. Implemente o diagrama de classes abaixo:



11. Crie uma classe chamada AuditoriaInterna que tenha dois métodos que tenham uma List de Tributáveis e os métodos:
  - a. `adicionar(Tributável);`
  - b. `calcularTributos();` retorna um double que representa a soma de todos os cálculos dos tributos de todos os tributáveis;
  - c. Crie uma classe de testes que instancie várias classes **ContaCorrente** e **SeguroDeVida**, adicione-as na classe **AuditoriaInterna** e exiba o resultado do método `calculaTributos`. Perceba que a classe de auditoria não se preocupa que tipo de classe está sendo passada.
12. Crie uma classe de testes que instancie várias classes **ContaCorrente** e **SeguroDeVida**, adicione-as na classe **AuditoriaInterna** e exiba o resultado do

método **calculaTributos**. Perceba que a classe de auditoria não se preocupa que tipo de classe está sendo passada.