



C . E . S . A . R

Pessoas impulsionando inovação.
Inovação impulsionando negócios.



Classes Abstratas e Interfaces



C.E.S.A.R



DEFINIÇÃO

O que é classe abstrata na programação orientada a objetos?

Pode-se dizer que as classes abstratas servem como “modelo” para outras classes que dela herdem. Este tipo de classe possui uma característica muito específica, que é o de não permitir que novos objetos sejam instanciados a partir desta classe. Por este motivo, as classes abstratas possuem o único propósito de servirem como superclasses a outras classes do Java.

Classe Abstrata

Para ter um objeto de uma classe abstrata é necessário criar uma classe mais especializada herdando dela e então instanciar essa nova classe. Os métodos da classe abstrata devem então serem sobrescritos nas classes filhas.

Classe Abstrata


Uma classe abstrata é desenvolvida para representar entidades e conceitos abstratos, sendo utilizada como uma classe pai, pois não pode ser instanciada. Ela define um modelo (template) para uma funcionalidade e fornece uma implementação incompleta - a parte genérica dessa funcionalidade - que é compartilhada por um grupo de classes derivadas. Cada uma das classes derivadas completa a funcionalidade da classe abstrata adicionando um comportamento específico.

Métodos Abstratos

Os métodos abstratos estão presentes somente em classes abstratas, e são aqueles que não possuem implementação. A sintaxe deste tipo de método é a seguinte:

abstract

MÉTODO ABSTRATO

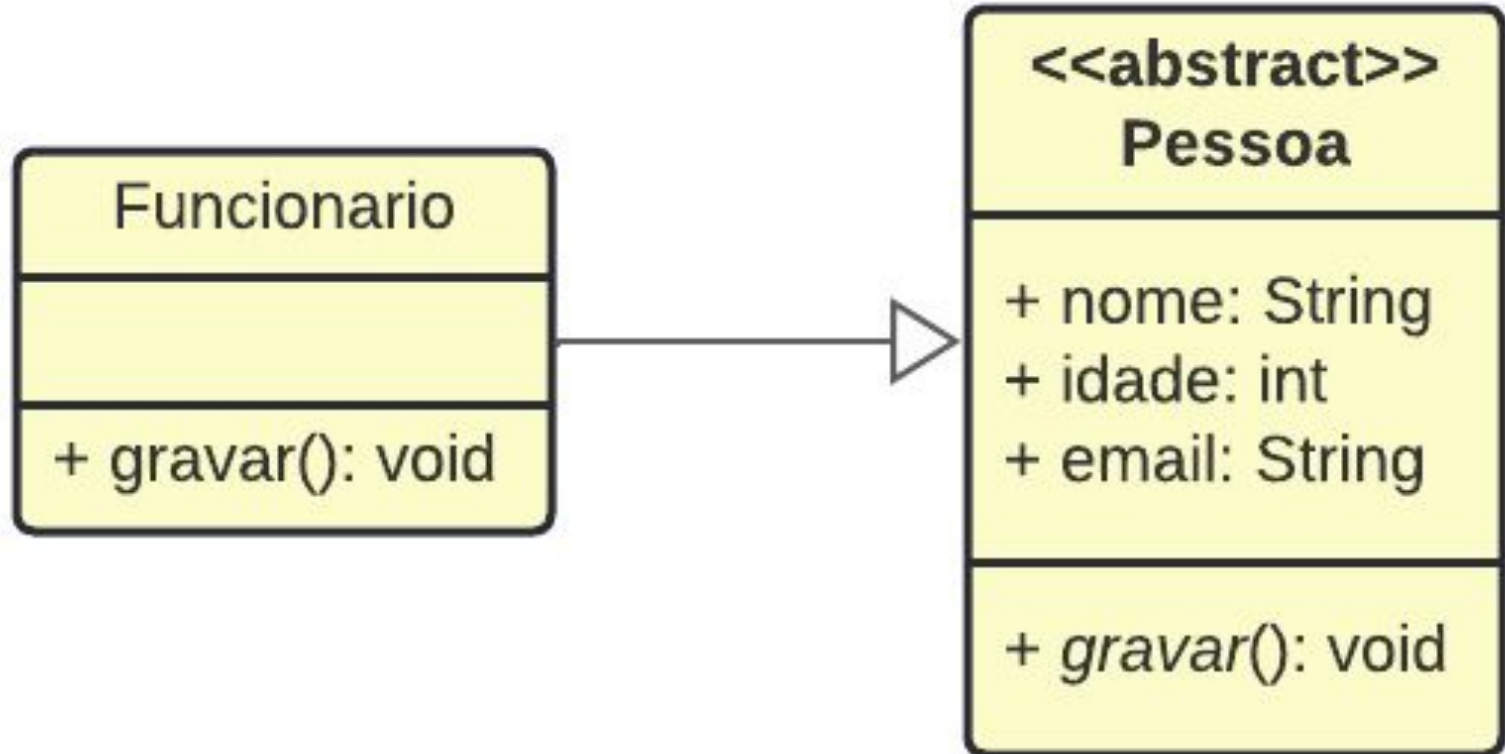


```
public abstract class Pessoa {  
    public String Nome;  
    public int Idade;  
    public String Email;  
  
    public abstract void gravar();  
}
```

MÉTODO ABSTRATO

```
public class Funcionario extends Pessoa {  
    @Override  
    public void gravar() {  
        Nome = "Gerson";  
        Idade = 30;  
        Email = "grs@cesar.school";  
    }  
}
```


DIAGRAMA DE CLASSE



Interfaces

Considerada uma Entidade, as interfaces têm papel fundamental no desenvolvimento de software orientado à objetos. Por várias vezes, nós precisamos, de alguma forma, especificar um conjunto de métodos que um grupo de classes deverá, obrigatoriamente, implementar. Para atingir este efeito, utilizamos as interfaces.

Utilizando interface em Java

Dentro das interfaces existem somente assinaturas de métodos e propriedades, cabendo à classe que a utilizará realizar a implementação das assinaturas, dando comportamentos práticos aos métodos.

INTERFACE

```
public interface FiguraGeometrica {  
    public String getNomeFigura();  
    public int getArea();  
    public int getPerimetro();  
}
```

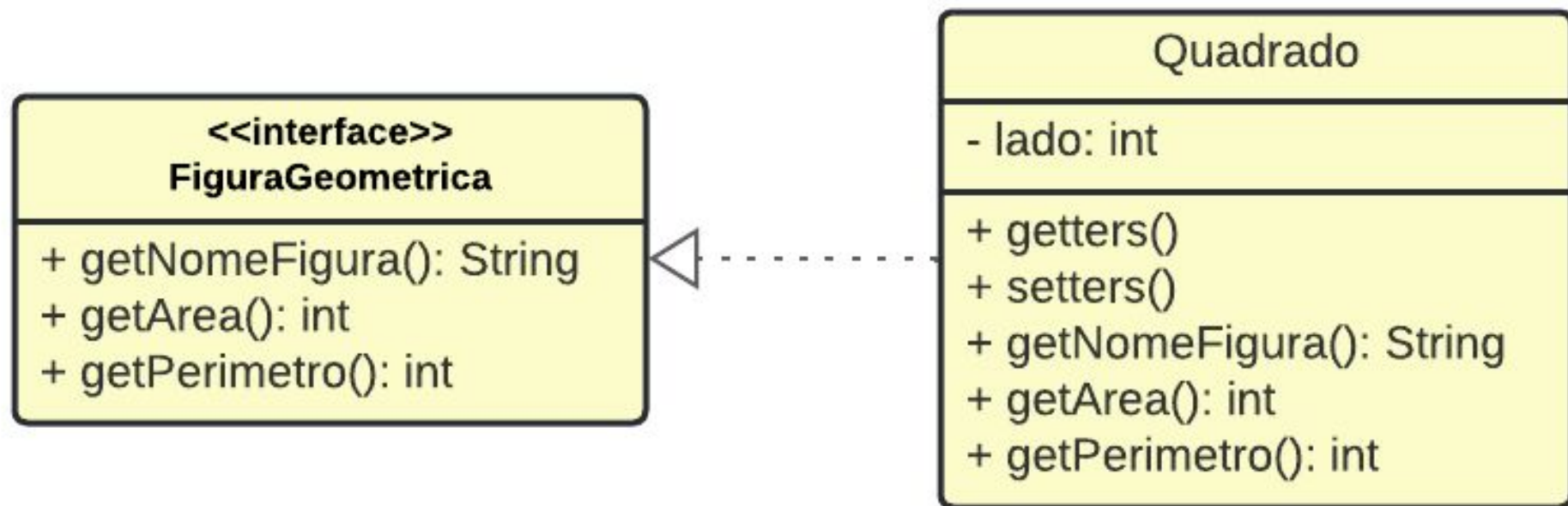
INTERFACE

```
public class Quadrado implements FiguraGeometrica {
    private int lado;
    public int getLado() {
        return lado;
    }
    public void setLado(int lado) {
        this.lado = lado;
    }
    @Override
    public int getArea() {
        int area = 0;
        area = lado * lado;

        return area;
    }
    @Override
    public int getPerimetro() {
        int perimetro = 0;

        perimetro = lado * 4;
        return perimetro;
    }
    @Override
    public String getNomeFigura() {
        return "quadrado";
    }
}
```

DIAGRAMA DE CLASSE



Qual a diferença entre classe abstrata e interface?

Basicamente, a interface não permite a inserção de qualquer tipo de código, muito menos se ele for padrão. Já a classe abstrata pode oferecer uma codificação completa, o padrão ou apenas possuir a declaração de um esqueleto para ser sobrescrita posteriormente.

Quando usar cada uma delas?

Interface

O ideal é utilizá-la quando várias classes diferentes compartilham somente a assinatura de seus métodos, a exemplo das funcionalidades oferecidas por uma determinada interface criada.

Podemos concluir que as interfaces estão aí para ditar o que uma classe deve fazer, ajudando a definir quais são as habilidades que cada classe assinante desse "contrato" deve possuir.

Quando usar cada uma delas?

Classe abstrata

Ao contrário da interface, que pode estar envolvida com diversas classes sem qualquer relação, uma classe abstrata ainda é uma classe.

Quando uma classe X herda de outra Y, é a mesma coisa que dizer que X é um Y. Neste caso, quando queremos desenvolver várias classes que compartilham um mesmo comportamento, a classe abstrata é a solução perfeita para basear a criação de todas elas, servindo como um molde para as suas derivadas.

Podemos entender que a classe abstrata define a identidade de suas derivadas, dando base para como elas devem se comportar, aumentando o acoplamento entre elas.

Exercícios

Exercício 1

1- Implemente, em Java, uma classe abstrata de nome `Quadrilatero` onde são declarados dois métodos abstratos:

- `float calcularArea();`
- `float calcularPerimetro();`

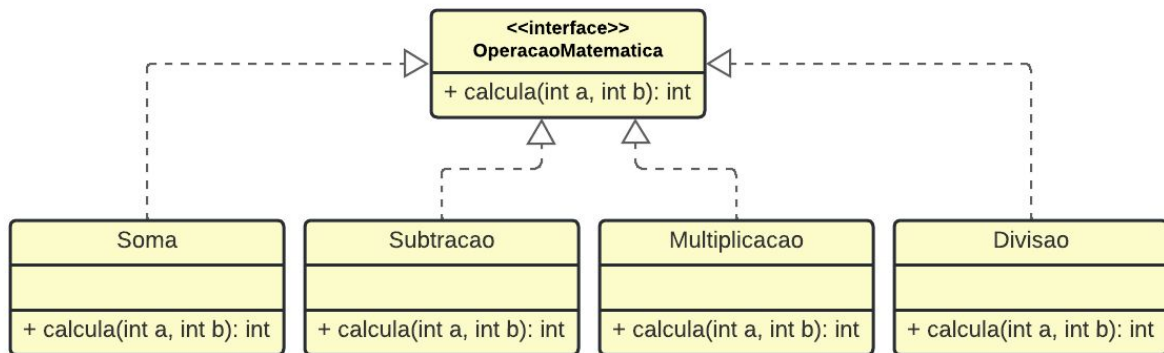
Crie, como subclasse de `Quadrilatero`, uma classe de nome `Retangulo` cujas instâncias são caracterizadas pelos atributos `lado` e `altura` ambos do tipo `float`.

Implemente na classe `Retangulo` os métodos herdados de `Quadrilatero` e outros que ache necessários.

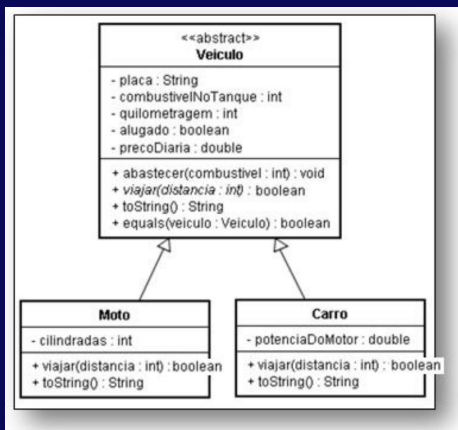
Exercício 2

2 - Implementar uma aplicação que declara uma Interface do tipo OperacaoMatematica que deve realizar uma operação matemática e imprimir o seu resultado, de acordo com o diagrama abaixo.

- OBS 1: Não defina a e b como atributos.
- OBS 2: Implemente um construtor padrão para cada uma das classes.



Exercício 3



3 - Implemente as classes conforme o diagrama:

- O método `abastecer` deve adicionar o valor passado por parâmetro ao atributo `combustivelNoTanque`
- O método `equals` deve retornar `true` se o valor do atributo `placa` for o mesmo para os dois objetos.
- Método `viajar` - Moto: o método deve considerar que uma moto faz 30km com 1 litro de combustível. Logo, deve verificar se o combustível no tanque é suficiente para percorrer a distância passada como parâmetro do método. Caso seja, deverá reduzir essa quantidade do atributo `combustivelNoTanque` e adicionar o valor da distância ao valor do atributo `quilometragem`. Retorne o valor `true` caso seja possível realizar a viagem. Caso contrário, retorne `false`.
- Método `viajar` - Carro: Deve considerar que um carro faz 10km com um litro de combustível. Fazer as mesmas operações que as descritas no método `viajar` da classe **Moto**.
- Na classe principal instancie objetos das classes implementadas e teste.



C . E . S . A . R

Pessoas impulsionando inovação.
Inovação impulsionando negócios.

NOSSO CONTATO

grs@cesar.school

