GSI511 – Programação orientada a objetos I

Profa. Alessandra Paulino

2º Período

Sumário

Classe abstrata I

Uma classe abstrata – ou **classe virtual** – é uma classe incompleta onde alguns ou todos os seus métodos não possuem implementação

Todas as classes vistas até este ponto não são abstratas, são *classes concretas*.

Classe abstrata II

Quando usamos herança, em diversas ocasiões as classes base são bastante *genéricas* (principalmente se houver *vários níveis de herança*);
Neste caso, pode-se implementar classes que definem *comportamentos genéricos* – as **classes abstratas**:

- A essência da superclasse é definida e pode ser *parcialmente* implementada;
- Detalhes são definidos em subclasses especializadas;
- **Não** podem ser instanciadas (servem apenas para reunir características comuns dos descendentes).

Java

Palavra reservada `abstract`

Classe abstrata III

Exemplo:

```
1 public abstract class Conta {
2     private int num;
3     private float saldo;
4 }
```

```
1 public class Poupanca extends Conta {
2     ...
3 }
```

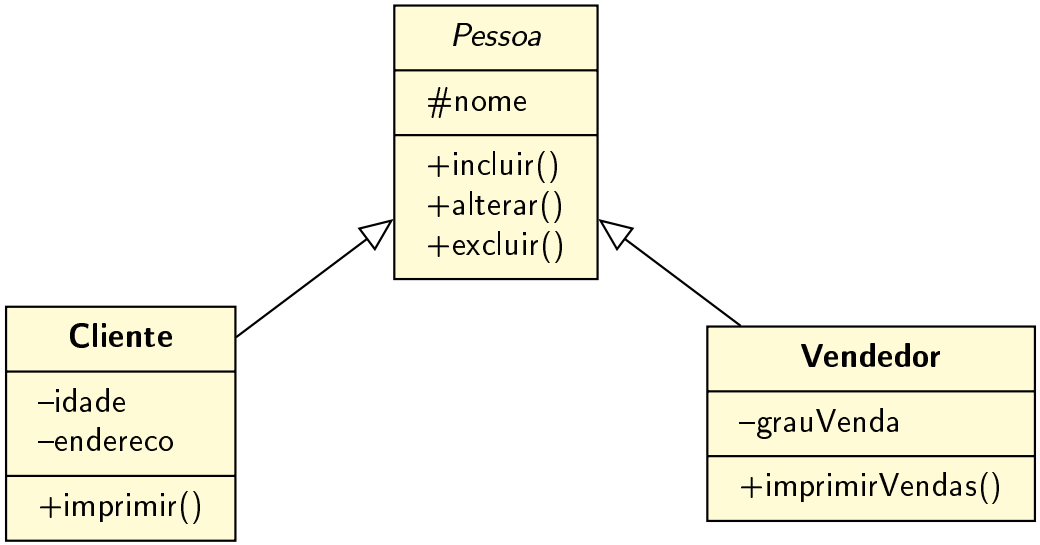
```
1 public class ContaEspecial extends Conta {
2     ...
3 }
```

Classe abstrata IV

```
1 Conta c; // esta linha está ok
2 c = new Conta(); // ERRO: não posso criar objeto
    de classe abstrata
3 c = new ContaEspecial(); // ok
```

Classe abstrata V

Outro exemplo (em UML, nome de classe abstrata é escrito em *itálico*):



- A classe `Pessoa` existe para reunir as características.
- Um objeto efetivo dentro de uma loja deve ser cliente ou vendedor. *Não existe apenas pessoa*.

Métodos abstratos I

Métodos abstratos são métodos definidos exclusivamente dentro de classes abstratas, mas *não são* implementados nas mesmas (apenas sua *assinatura* é especificada).

- Os métodos abstratos devem ser **obrigatoriamente** implementados em **toda classe herdeira (concreta)** da classe abstrata em que são definidos.

Declarar um método como abstrato é uma forma de obrigar o programador a redefinir esse método em todas as subclasses para as quais se deseja criar objetos.

Exemplo:

```java
public abstract class Forma {
    public abstract double perimetro();
    public abstract double area();
}
```

```java
public class Circulo extends Forma {
    public double perimetro() {
        return 2.0*Math.PI*this.raio;
    }
    public double area() {
        return Math.PI*Math.pow(this.raio,2.0);
    }
}
```

Métodos abstratos III

```
1 public class Quadrado extends Forma {
2     public double perimetro() {
3         return 4.0*this.lado;
4     }
5     public double area() {
6         return Math.pow(this.lado,2.0);
7     }
8 }
```

Exercício I

Implemente a hierarquia de classes Empregado (superclasse), Chefe, PorComissao, PorItem e PorHora (subclasses), seguindo as orientações abaixo.

- Todas as classes devem conter construtor com parâmetros e método para retornar uma String com as informações do objeto.
- Empregado deve ser classe abstrata, conter atributos nome e sobrenome, e conter método abstrato retornarSalario (este deve retornar o salário de acordo com o cálculo específico por tipo de empregado).

Exercício II

- Chefe: salário fixo e predefinido.
- PorComissao: valor fixo + comissão * vendas.
- PorItem: valor por produção * quantidade produzida.
- PorHora: valor por hora * total de horas trabalhadas.
- Programa principal que cria um vetor de Empregados e instancia diversos tipos de empregados, mostra na tela as informações de todos os empregados e a soma total dos salários de todos os empregados cadastrados.

Referências

1. Addison-BOOCH, G., RUMBAUGH, J., JACOBSON, I. *UML, Guia do Usuário*. Rio de Janeiro: Campus, 2000.
2. FOWLER, M. *UML Essencial*, 2a Edição. Porto Alegre: Bookman, 2000.
3. LARMAN, C. *Utilizando UML e Padrões: Uma Introdução à Análise e ao Projeto Orientado a Objetos*. Porto Alegre: Bookman, 2001.

Os slides dessa apresentação foram cedidos por:

- Graça Marietto e Francisco Zampirolli, UFABC
- Profa Katti Faceli, UFSCar/Sorocaba
- Marcelo Z. do Nascimento, FACOM/UFU

LaTeXagem e adaptações: Renato Pimentel, FACOM/UFU