FCT/Unesp – Presidente Prudente
Departamento de Matemática e Computação

Programação Orientada a Objetos

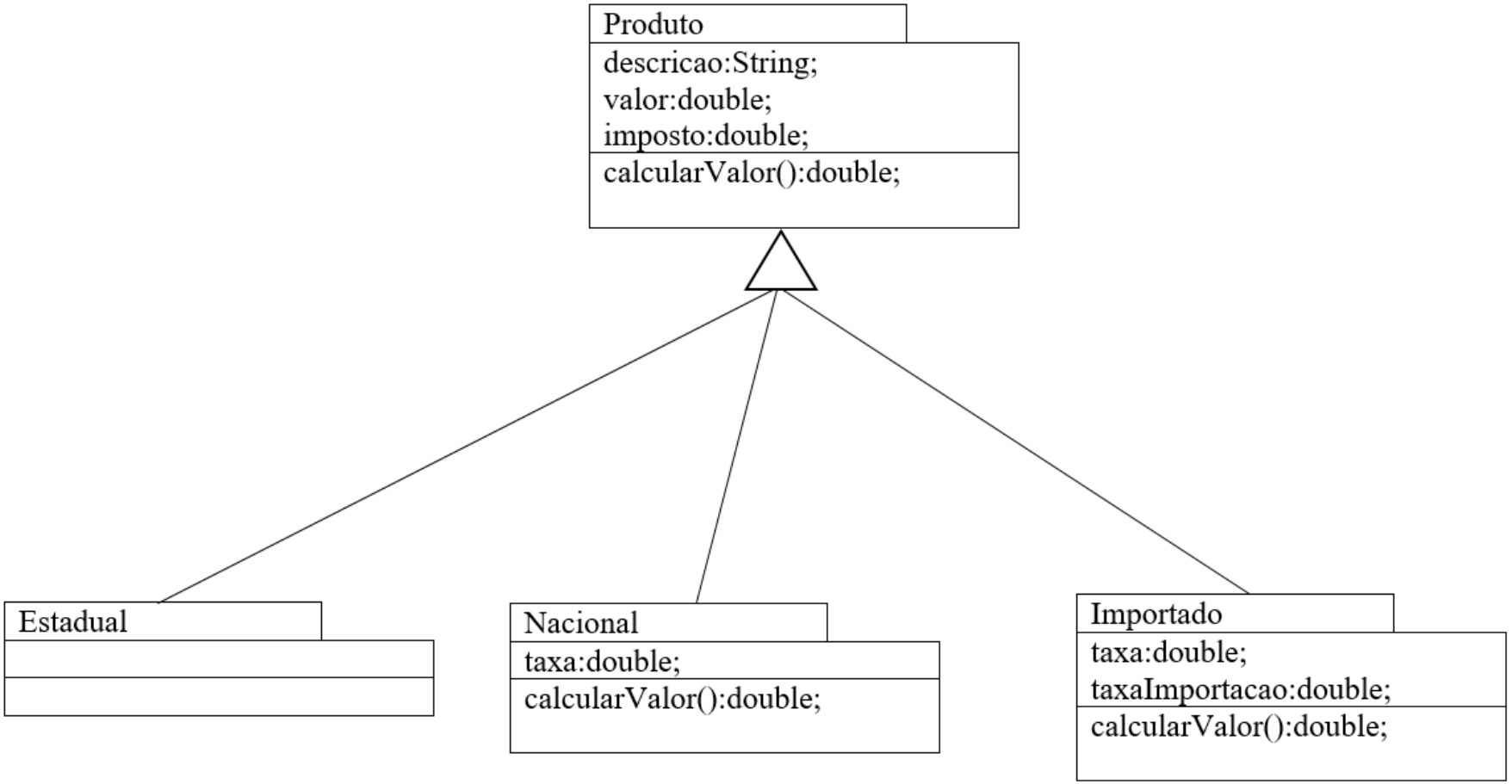
Aula 6 – Classes Abstratas

Prof. Danilo Medeiros Eler

danilo.eler@unesp.br

Departamento de Matemática e Computação
Faculdade de Ciências e Tecnologia
Unesp – Universidade Estadual Paulista
Presidente Prudente/SP, Brasil

Aula anterior: Herança



Classes e Métodos Abstratos

- Uma classe abstrata não pode ser instanciada
  - Representa uma classe de objetos e define alguns comportamentos
- Uma classe abstrata não implementa alguns métodos (ou todos os métodos), servindo como uma base para as subclasses

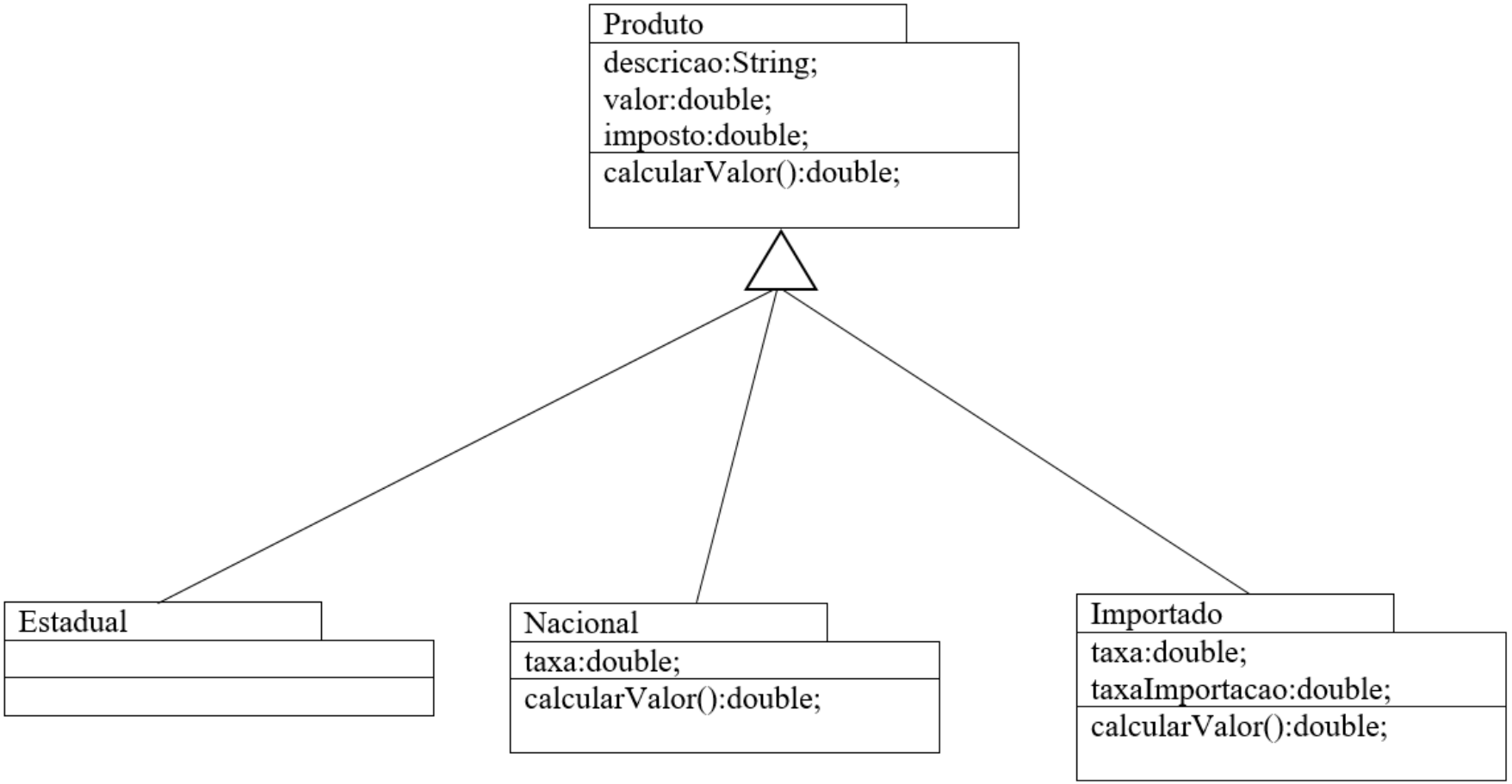
Classes e Métodos Abstratos

- Uma classe abstrata pode forçar um comportamento nas subclasses por meio de métodos abstratos

- Um método abstrato indica que a classe não implementa aquele método
  - Ele deve ser obrigatoriamente implementado nas subclasses

- Todos os métodos não definidos de uma classe abstrata deverão ser definidos nas subclasses ou estas deverão a ser consideradas abstratas e não poderão ser instanciadas

Classes e Métodos Abstratos



Classes e Métodos Abstratos

- Para definir que uma classe é abstrata basta adicionar *abstract* em sua codificação

```
public abstract class Produto {
    protected String descricao;
    protected double valor;
    protected double imposto;

    public Produto() {
        valor = 0;
        imposto = 10;
    }
```

Classes e Métodos Abstratos

- Objetos não podem ser instanciados a partir de classes abstratas

```
public static void main(String[] args) {
    // TODO code application logic here
    Produto p = new Produto();
}
```

Classes e Métodos Abstratos

- Objetos não podem ser instanciados a partir de classes abstratas

```
public static void main(String[] args) {
    // TODO code application logic here
    Produto p = new Produto();
}
```

Produto is abstract; cannot be instantiated

(Alt-Enter shows hints)

Classes e Métodos Abstratos

- Métodos abstratos podem definir comportamentos e deverão ser implementados nas subclasses
  - Um método abstrato é aquele que não tem corpo

```
public abstract class Produto {
    protected String descricao;
    protected double valor;
    protected double imposto;
    public Produto() {
        valor = 0;
        imposto = 10;
    }
    public abstract double valorFinal();
```

Classes e Métodos Abstratos

- A subclasse será forçada a implementar esse método ou terá que ser considerada abstrata

```
public class Estadual extends Produto{
    |
}
```

Classes e Métodos Abstratos

- A subclasse será forçada a implementar esse método ou terá que ser considerada abstrata

```
Estadual is not abstract and does not override abstract method valorFinal() in Produto
----
(Alt-Enter shows hints)
```

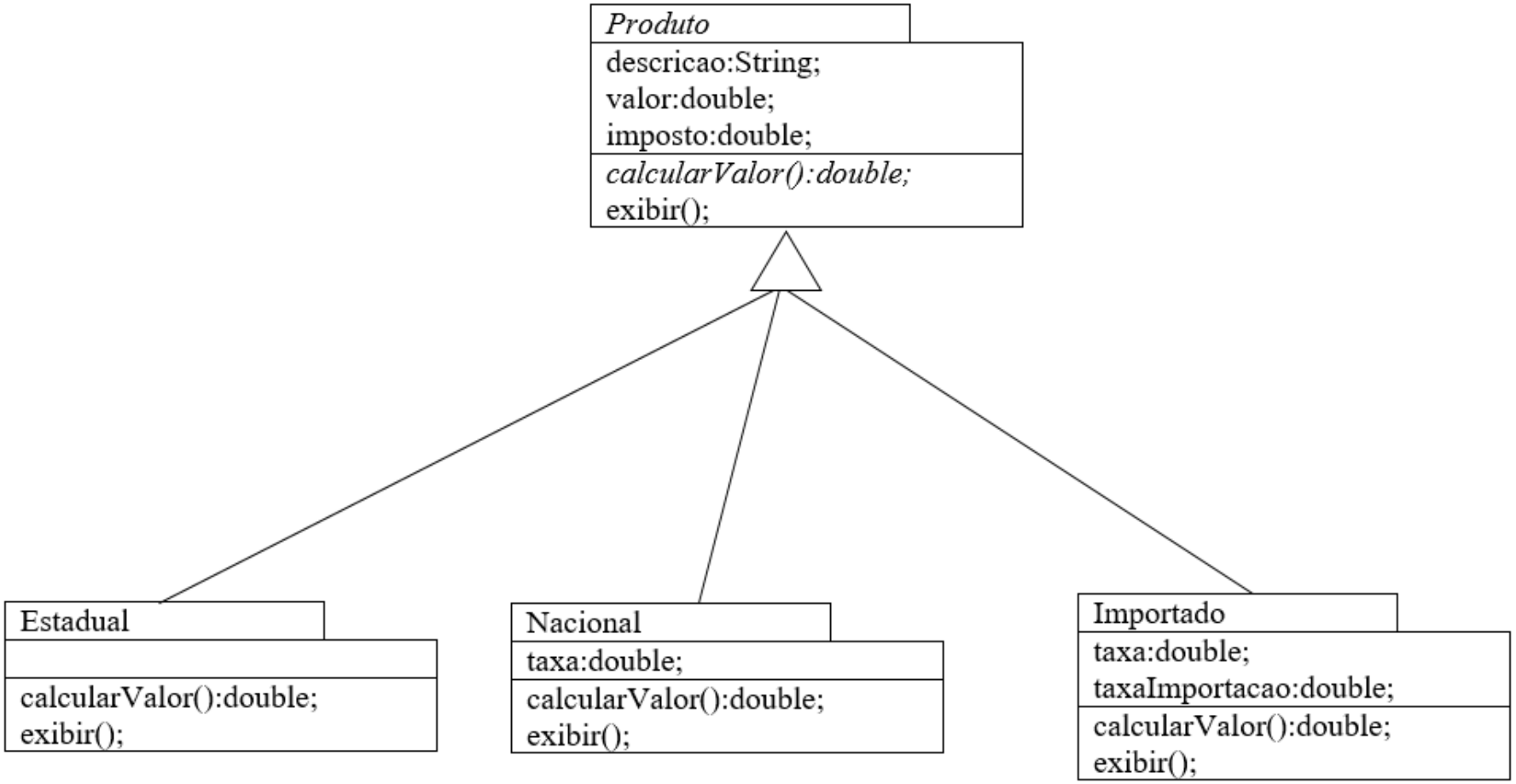
```
public class Estadual extends Produto{

}
```

Classes e Métodos Abstratos

- A subclasse será forçada a implementar esse método ou terá que ser considerada abstrata

```
public class Estadual extends Produto{
    @Override
    public double valorFinal() {
        return valor + valor * imposto / 100.0f;
    }
}
```

Classes e Métodos Abstratos



Classes e Métodos Abstratos

```
public abstract class Produto {
    protected String descricao;
    protected double valor;
    protected double imposto;

    public Produto() {
        valor = 0;
        imposto = 10;
    }

    public abstract double valorFinal();

    public void exibir() {
        System.out.println("Descrição: " + descricao);
        System.out.println("Valor: R$ " + valor);
        System.out.println("Imposto: " + imposto + "%");
        System.out.println("Valor Final: R$ " + valorFinal());
    }
```

Classes e Métodos Abstratos

```
public abstract class Produto {
    protected String descricao;
    protected double valor;
    protected double imposto;

    public Produto() {
        valor = 0;
        imposto = 10;
    }

    public abstract double valorFinal();

    public void exibir();
```

Classes e Métodos Abstratos

```
public abstract class Produto {
    protected String descricao;
    protected double valor;
    protected double imposto;

    public Produto() {
        valor = 0;
        imposto = 10;
    }

    public ab

    public void exibir();
```

missing method body, or declare abstract

(Alt-Enter shows hints)

Classes e Métodos Abstratos

```
public abstract class Produto {
    protected String descricao;
    protected double valor;
    protected double imposto;

    public Produto() {
        valor = 0;
        imposto = 10;
    }

    public abstract double valorFinal();

    public abstract void exibir();
```

Classes e Métodos Abstratos

- Todas as subclasses terão que implementar o método exibir

```
public class Estadual extends Produto{
    @Override
    public double valorFinal() {
        return valor + valor * imposto / 100.0f;
    }
}
```

Classes e Métodos Abstratos

- Todas as subclasses terão que implementar o método exibir

```
Estadual is not abstract and does not override abstract method exibir() in Produto
----
(Alt-Enter shows hints)
```

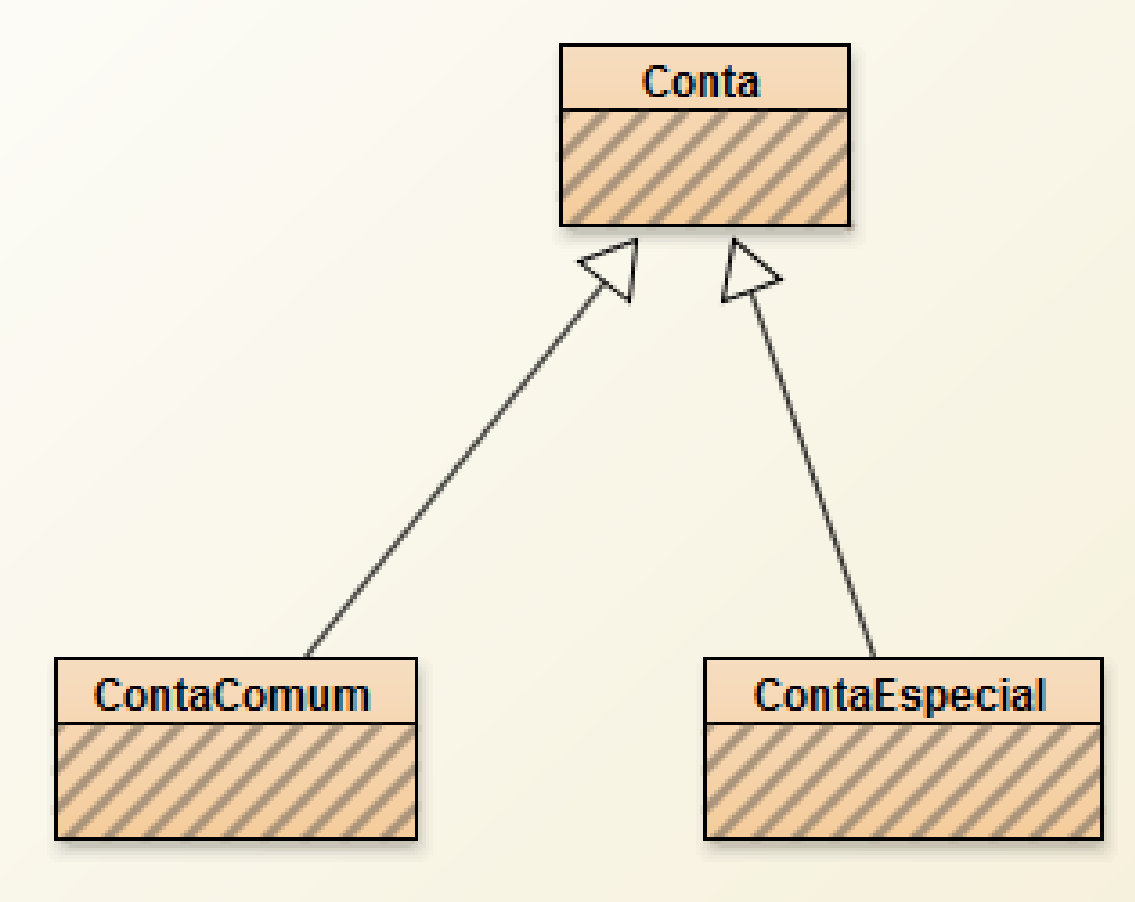
```
public class Estadual extends Produto{
    @Override
    public double valorFinal() {
        return valor + valor * imposto / 100.0f;
    }
}
```

Classes e Métodos Abstratos

- Todas as subclasses terão que implementar o método exibir

```
public class Estadual extends Produto{
    @Override
    public double valorFinal() {
       return valor + valor * imposto / 100.0f;
    }
    @Override
    public void exibir() {
        System.out.println("Descrição: "+descricao);
        System.out.println("Valor: R$ "+valor);
        System.out.println("Imposto: "+imposto);
        System.out.println("Valor Final: R$ "+valorFinal());
    }
}
```

Classes e Métodos Abstratos



Classes e Métodos Abstratos

```
public abstract class Conta{
    protected String cliente;
    protected double saldo;
    public Conta(){ saldo = 0 };
    public Conta(String cliente, double saldo){
      this.cliente = cliente;
      this.saldo = saldo;
   }
   public void depositar(double valor){
       saldo = saldo + valor;
   }
   public abstract void sacar();
   public abstract void relatorio();
}
```

Métodos abstratos que deverão ser implementados pelas subclasses

Classes e Métodos Abstratos

```
public ContaComum extends Conta{
    public ContaComum(){
        super();
    }
    public ContaComum(String cliente, double saldo){
        super(cliente, saldo);
    }
}
```

A classe ContaComum não é abstrata e, portanto, deve implementar os métodos abstratos definidos na classe Conta

Classes e Métodos Abstratos

```
public ContaComum extends Conta{
     public ContaComum(){
            super();
     }
     public ContaComum(String cliente, double saldo){
          super(cliente, saldo);
    }
@Override
   public void sacar(double valor){
          .............
   }
@Override
   public void relatorio(){
     .............
   }
}
```

Métodos que devem ser obrigatoriamente implementados pela subclasse ContaComum

Classes e Métodos Abstratos

```
public ContaEspecial extends Conta{
     protected double limite;
     public ContaEspecial(){
            super();
            limite = 0;
     }
     public ContaEspecial(String cliente, double saldo, double limite){
          super(cliente, saldo);
          this.limite = limite
    }
@Override
   public void sacar(double valor){
          .............
   }
@Override
   public void relatorio(){
     .............
   }
}
```

Métodos que devem ser obrigatoriamente implementados pela subclasse ContaEspecial

Classes e Métodos Abstratos

- Exemplo prático

Referências

BIBLIOGRAFIA BÁSICA

1. SINTES, A., Aprenda programação orientada a objetos em 21 dias, Pearson Education do Brasil, 2002.
2. VAREJÃO, F., Linguagens de programação : Java, C e C++ e outras : conceitos e técnicas, Campus, 2004.
3. DEITEL, H. M., DEITEL, P. J., **Java:** como programar, São Paulo: Pearson Education do Brasil, 2010. 1144p.
4. DEITEL, H. M., DEITEL, P. J., **Java:** como programar, Porto Alegre: Bookman, 2003. 1386p.
5. SAVITCH, W. J., C++ absoluto, Pearson Education : Addison Wesley, 2004.

Capítulo 9

BIBLIOGRAFIA COMPLEMENTAR

1. BERMAN, A. M. ***Data Structures via C++: Objects by Evolution***, Oxford University Press Inc., 1997.
2. BARNES, D.J. & KÖLLING, M., Programação orientada a objetos com Java, Pearson Education : Prentice Hall, 2004.
3. DEITEL, H. M. e DEITEL, P. J. ***C++: Como Programar***, Bookman, 2001.
4. GILBERT, R. F. e FOROUZAN, B. A. ***Data Structures: A Pseudo Approach with C++***, Brooks/Cole Thomson Learning, 2001.
5. MUSSER, D. R. e SAINI, A. STL ***Tutorial and Reference Guide: Programming with the Standard Template Library***, Addison-Wesley, 1996.
6. SEBESTA, R. W. ***Conceitos de Linguagem de Programação***, 4ª Ed., Bookman, 2003.
7. SEDGEWICK, R. ***Algorithms in C++***, Addison-Wesley, 2002.
8. STROUSTRUP, B. ***A Linguagem de Programação C++,*** 3ª Ed., Bookman, 2000.

Links

- http://www.dca.fee.unicamp.br/cursos/PooJava/Aulas/poojava.pdf
- http://www.di.ubi.pt/~pprata/poo_10_11.htm
- https://sites.google.com/site/fpaulovich/Home/Teaching/scc-604-programacao-orientada-a-objetos

FCT/Unesp – Presidente Prudente
Departamento de Matemática e Computação

Programação Orientada a Objetos

Aula 6 – Classes Abstratas

Prof. Danilo Medeiros Eler

danilo.eler@unesp.br

Departamento de Matemática e Computação
Faculdade de Ciências e Tecnologia
Unesp – Universidade Estadual Paulista
Presidente Prudente/SP, Brasil