



Classe Abstrata

Prof^a. Rachel Reis
rachel@inf.ufpr.br



Problema

- Uma empresa de tecnologia contrata empregados por meio de uma das seguintes modalidades de pagamento:



mensalista



horista



Problema

- Uma empresa de tecnologia contrata empregados por meio de uma das seguintes modalidades de pagamento:



mensalista

- Salário mensal fixo + horas extras.
- Hora-extra = $1/160$ salário mensal.



Problema

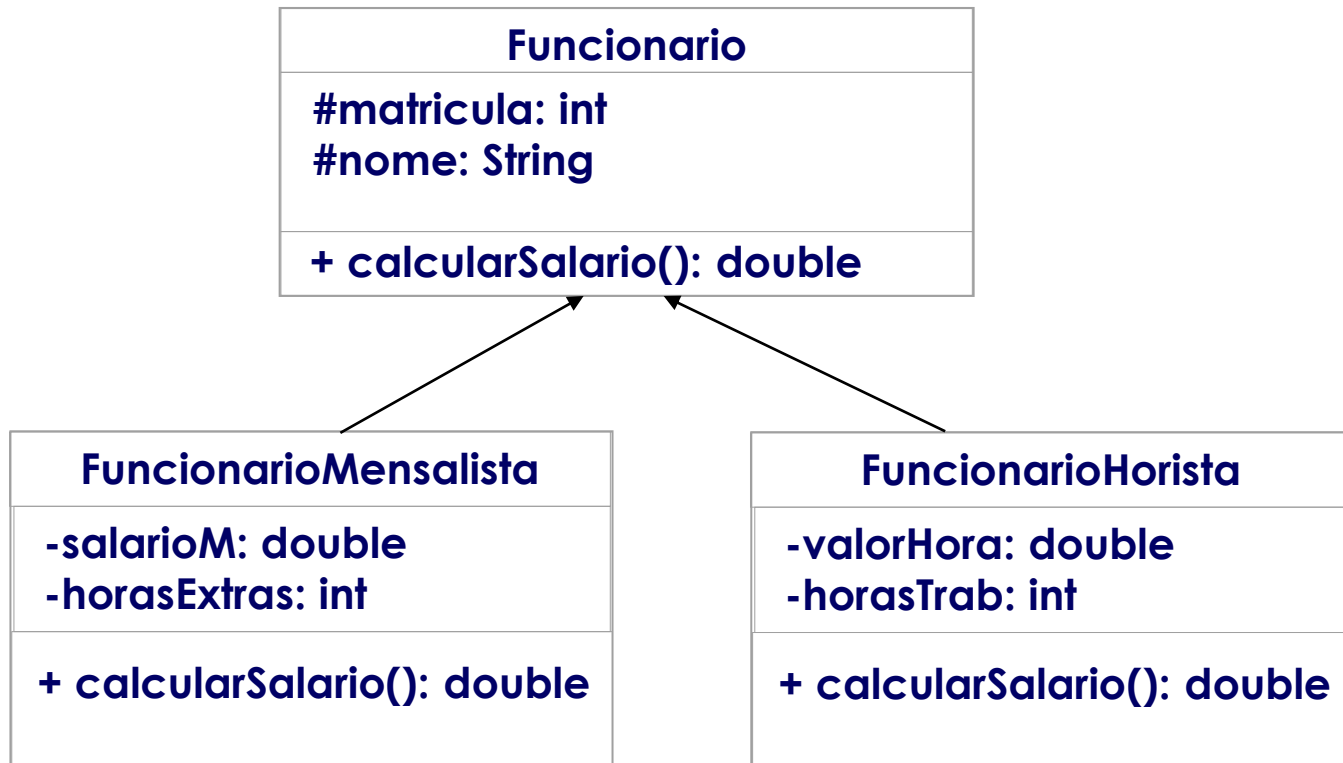
- Uma empresa de tecnologia contrata empregados por meio de uma das seguintes modalidades de pagamento:



horista

- Horas trabalhadas no mês.
- Não existe hora-extra.

Solução



```
public class Funcionario
{
    // Atributos
    protected int matricula;
    protected String nome;

    // Métodos get e set
    ...

    // Outros métodos
    public double calcularSalario() {
        ...
    }
}
```

Funcionario

#matricula: int

#nome: String

+ calcularSalario(): double

```
public class FuncionarioMensalista extends Funcionario
{
    // Atributos
    private double salarioM;
    private int horasExtras;

    // Métodos get e set
    ...

    // Outros métodos
    public double calcularSalario() {
        double val = this.horasExtras*(this.salarioM/160);
        double salarioT = this.salarioM + val;
        return salarioT;
    }
}
```

FuncionarioMensalista
-salarioM: double -horasExtras: int
+ calcularSalario(): double

```
public class FuncionarioHorista extends Funcionario
{
    // Atributos
    private double valorHora;
    private int horasTrab;

    // Métodos get e set
    ...

    // Outros métodos
    public double calcularSalario() {
        double salarioT = this.valorHora * this.horasTrab;
        return salarioT;
    }
}
```

FuncionarioHorista
-valorHora: double -horasTrab: int
+ calcularSalario(): double


```
public class Principal{  
    public static void main(String args[]){  
  
        FuncionarioMensalista obj1 = new  
                                FuncionarioMensalista();  
  
        obj1.setNome("Flash");  
        obj1.setMatricula(123);  
        obj1.setSalarioM(1000.0);  
        obj1.setHorasExtra(10);  
  
        S.o.p("Salario: R$" + obj1.calcularSalario());  
    }  
}
```



- 123
- Flash
- 1000.0
- 10

→ obj1



```
public class Principal{  
    public static void main(String args[]){  
  
        FuncionarioHorista obj2 = new  
                                FuncionarioHorista();  
  
        obj2.setNome("Hulk");  
        obj2.setMatricula(456);  
        obj2.setValorHora(90.0);  
        obj2.setHorasTrab(100);  
  
        S.o.p("Salario: R$" + obj2.calcularSalario());  
    }  
}
```



- 456
- Hulk
- 90.0
- 100

→ obj2

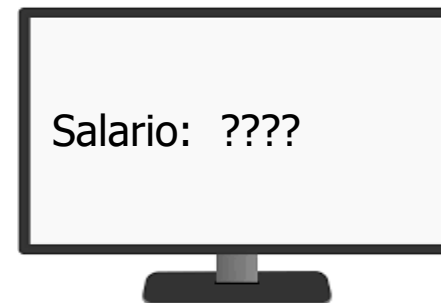


```
public class Principal{  
    public static void main(String args[]) {  
  
        Funcionario obj3 = new Funcionario();  
  
        obj3.setNome("Aquaman");  
        obj3.setMatricula(789);  
  
        S.o.p("Salario: R$" + obj3.calcularSalario());  
    }  
}
```



- 789
- Aquaman

→ obj3

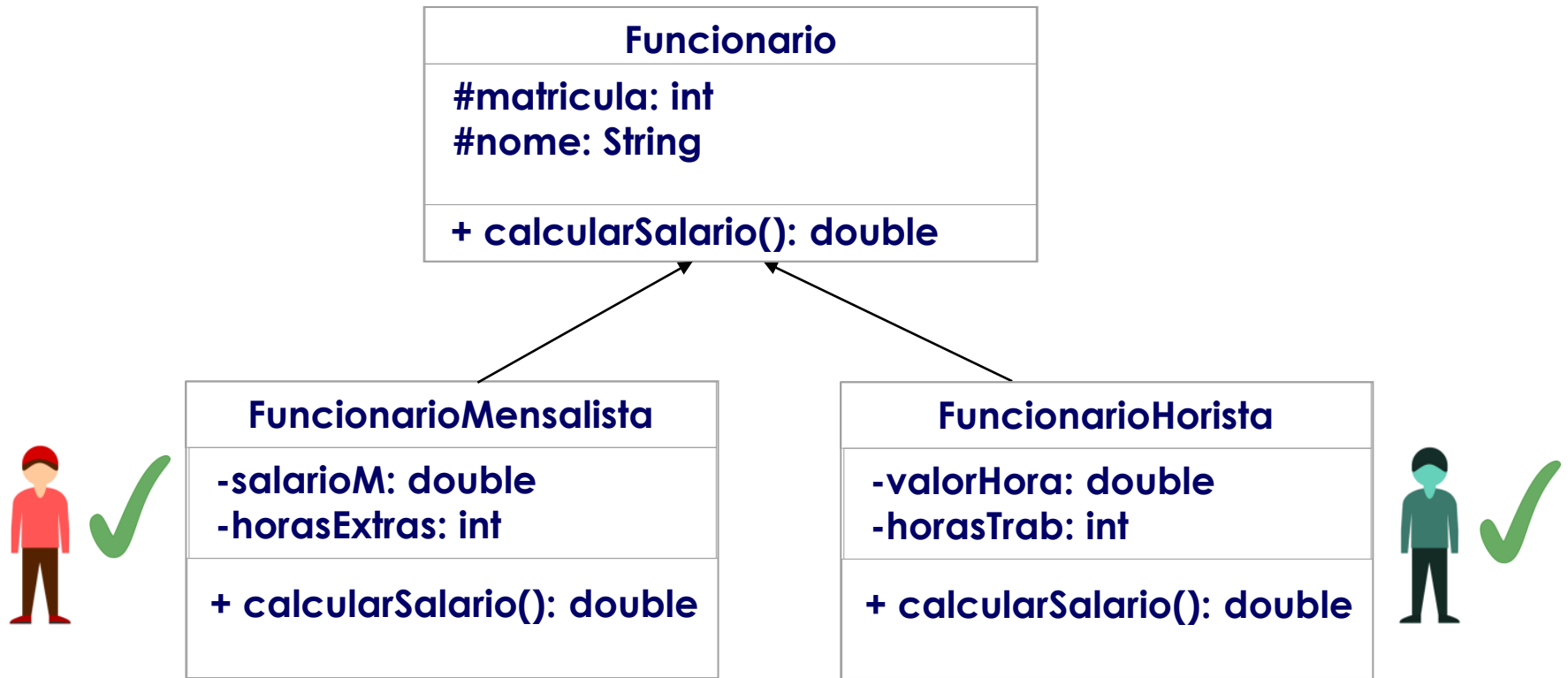


Faz sentido instanciar um objeto da classe Funcionário?

Funcionario
#matricula: int #nome: String
+ calcularSalario(): double

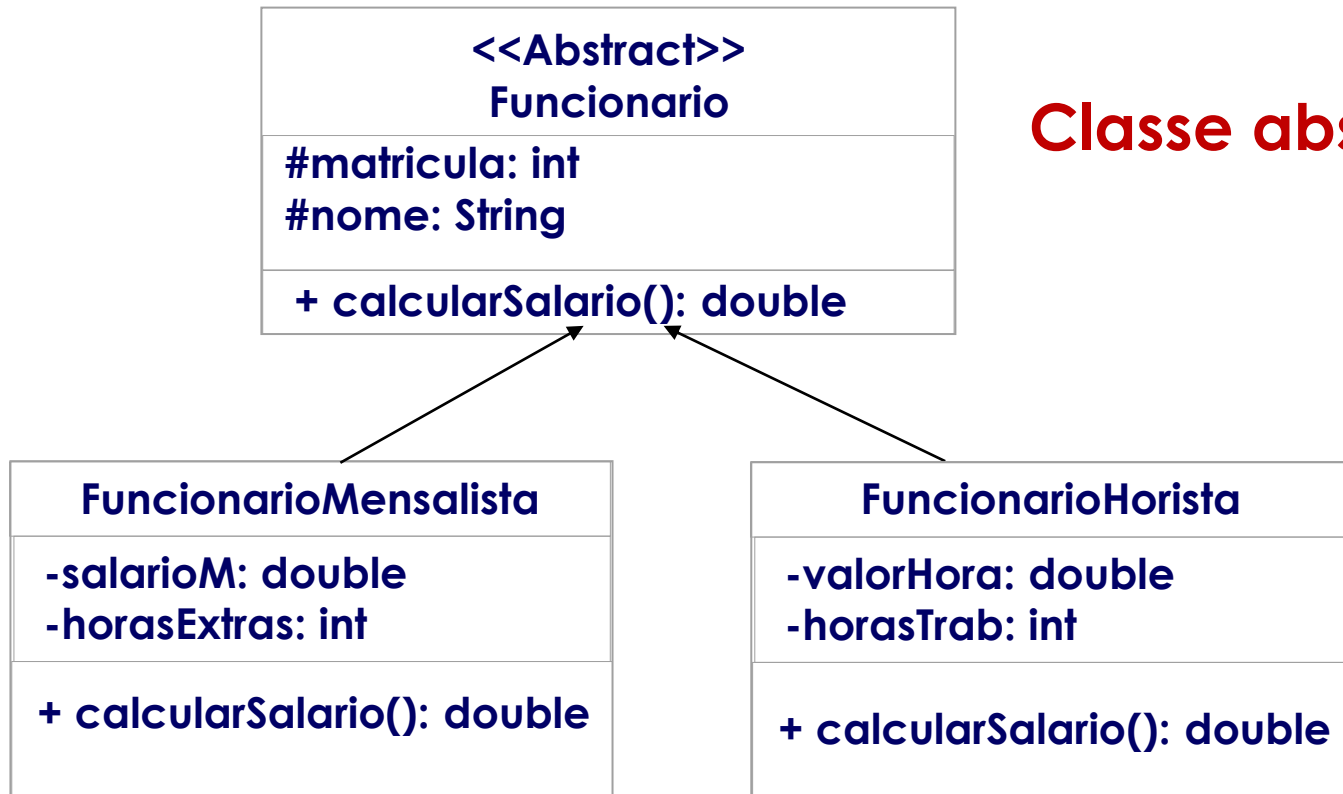


- Nenhum empregado jamais será instanciado na superclasse Funcionario.



- Objetos serão criados somente a partir das subclasses.

Solução



Classe abstrata

Classes concretas



Classe Abstrata – o que é?

- Uma superclasse extremamente genérica, que não pode ser instanciada.

```
Funcionario objeto1 = new Funcionario();
```





Classe Abstrata – o que é?

- Formada por métodos:
 - Com implementação:

```
public double calcularSalario() {  
    // implementação do método  
}
```

- Sem implementação (chamados de métodos abstratos):

```
public abstract double calcularSalario();
```




Classe Abstrata – o que é?

- É um meio termo entre *classe* e *interface*
 - Classe: possui implementação para todos os métodos declarados.
 - Interface: não possui nenhum método implementado.



Classe Abstrata – como usar

- Para declarar uma classe como abstrata deve-se usar a palavra-chave *abstract*:

```
public abstract class Funcionario{}
```

- Uma classe abstrata contém zero ou mais métodos abstratos.

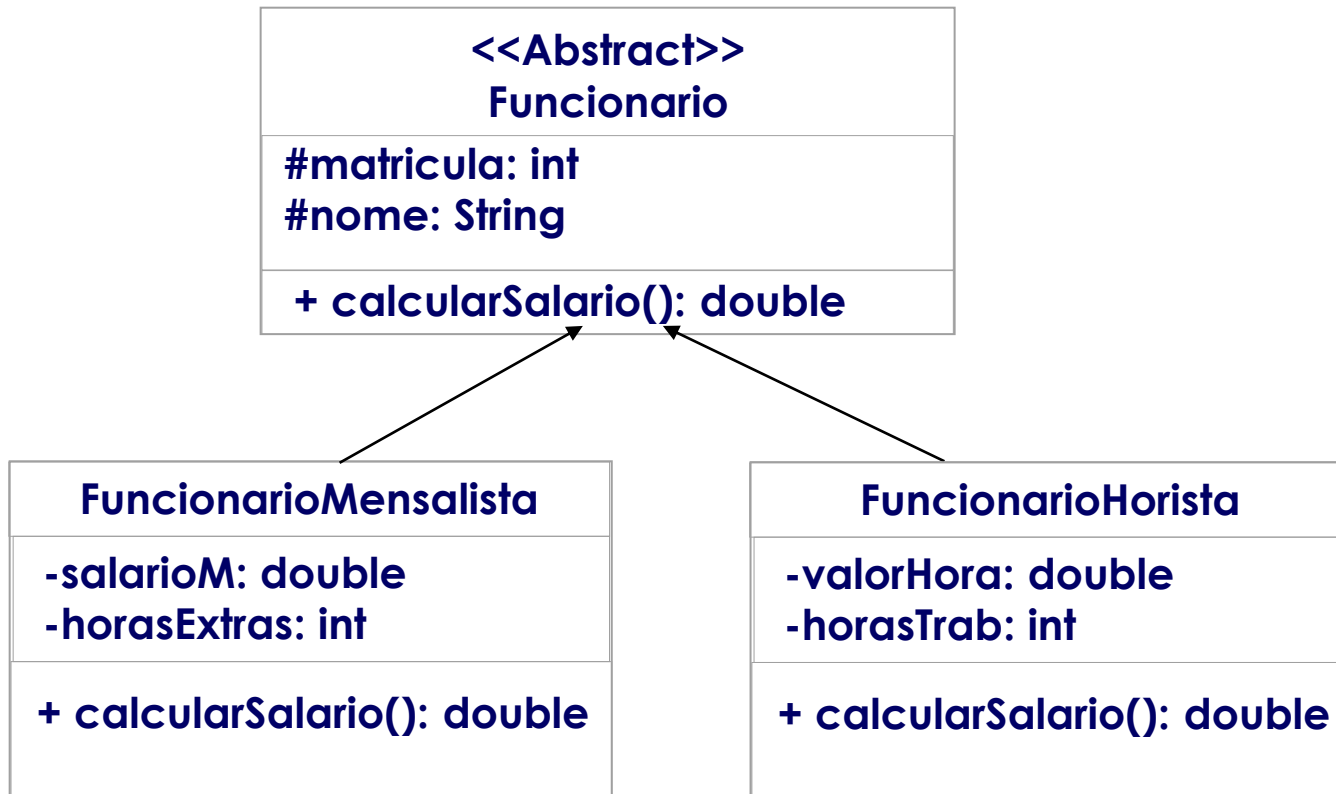


Classe Abstrata – como usar

- Um método abstrato é definido com a palavra-chave *abstract*.

```
public abstract double calcularSalario();
```

- O método abstrato define apenas a assinatura do método e, portanto, não contém código (veja que não há chaves e corpo do método).





- Vamos revisar a implementação da classe Funcionario.

```
public abstract class Funcionario
{
    // Atributos
    protected int matricula;
    protected String nome;

    // Métodos get e set
    ...

    // Outros métodos
    public abstract double calcularSalario();
}
```





Classe Abstrata – como usar

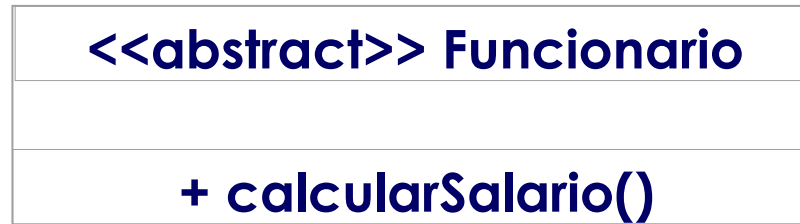
- Atributos e Construtores podem ser *abstract*?



No!

Se não é abstrata, implemente!

Classe abstrata

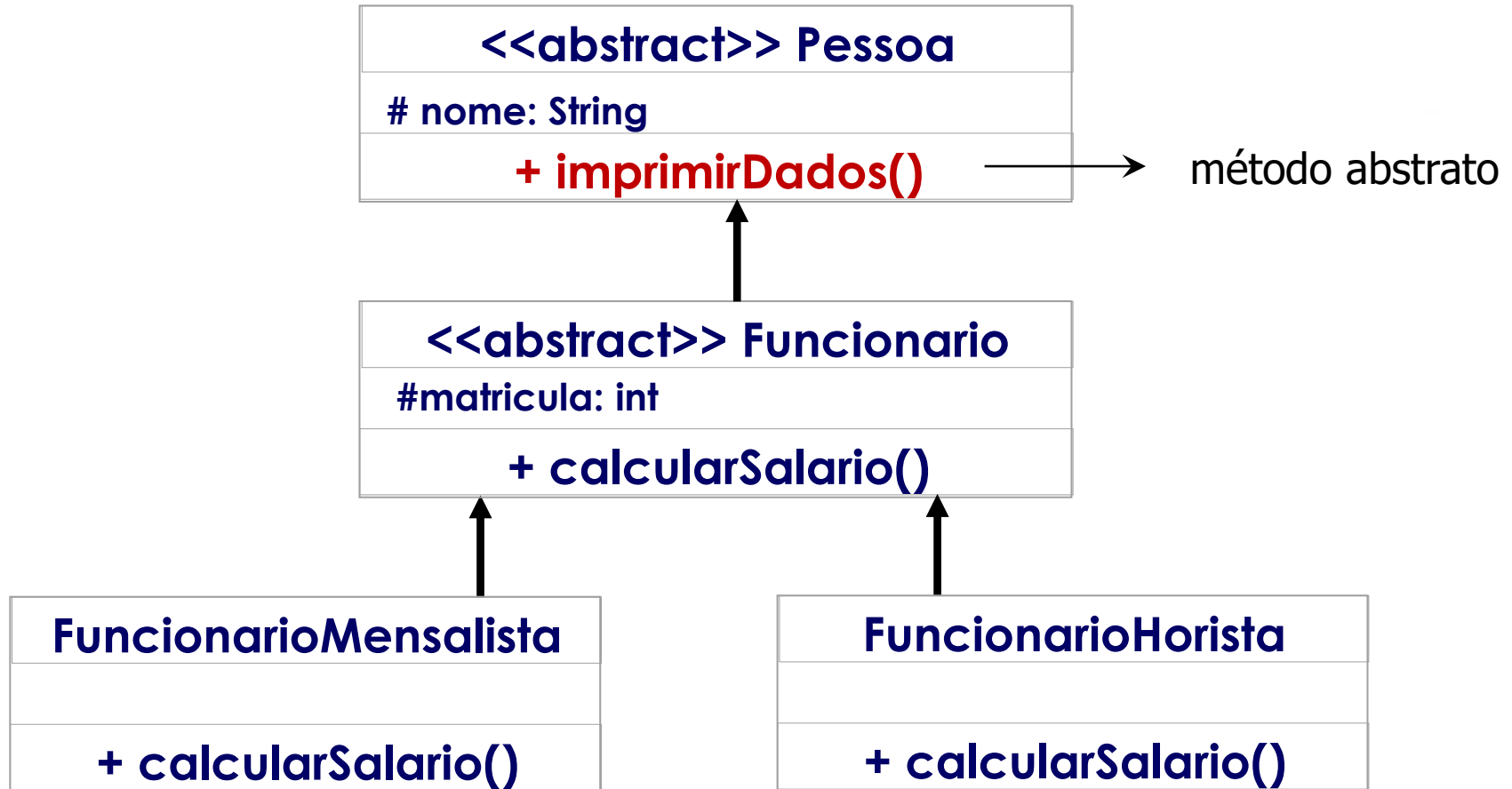


Classe concreta



- Uma classe concreta (não abstrata), que estende uma classe abstrata, deve fornecer a implementação dos métodos abstratos.

É possível herdar método abstrato?






É possível herdar método abstrato?

- Herdar método abstrato ou não
 - A primeira classe concreta (não abstrata) que herdar essa hierarquia tem que obrigatoriamente fornecer a implementação dos métodos abstratos herdados.
 - A primeira classe concreta (não abstrata) que herdar esse hierarquia tem que obrigatoriamente fornecer a implementação dos métodos abstratos herdados.

```
public abstract class Pessoa
{
    // Atributos
    protected String nome;

    // Métodos get e set
    ...



    // Outros métodos
    public abstract void imprimirDados();
}
```



```
public abstract class Funcionario extends Pessoa
{
    // Atributos
    protected int matricula;
protected String nome;

    // Métodos get e set
    ...

    // Outros métodos
    public abstract double calcularSalario();
}
```



```
public class FuncionarioMensalista extends Funcionario{
    // Atributos
    private double salarioM;
    private int horasExtras;

    // Métodos get e set
    ...

    // Outros métodos
    public double calcularSalario(){...}

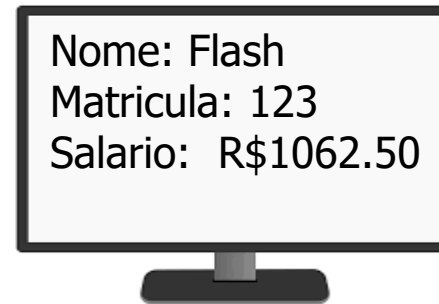
    ➡ public void imprimirDados() {
        System.out.println("Nome: " + this.nome);
        System.out.println("Matricula: " + this.matricula);
        System.out.println("Salario: " + this.calcularSalario());
    }
}
```

```
public class Principal{  
    public static void main(String args[]){  
  
        FuncionarioMensalista obj1 = new  
                                FuncionarioMensalista();  
  
        obj1.setNome("Flash");  
        obj1.setMatricula(123);  
        obj1.setSalarioM(1000.0);  
        obj1.setHorasExtras(10);  
  
        obj1.imprimirDados();  
  
    }  
}
```



- 123
- Flash
- 1000.0
- 10

→ obj1



```
public class FuncionarioHorista extends Funcionario{
    // Atributos
    private double valorHora;
    private int horasTrab;

    // Métodos get e set
    ...

    // Outros métodos
    public double calcularSalario(){...}

    ➡ public void imprimirDados() {
        System.out.println("Nome: " + this.nome);
        System.out.println("Matricula: " + this.matricula);
        System.out.println("Salario: " + this.calcularSalario());
    }
}
```

```
public class Principal{  
    public static void main(String args[]){  
  
        FuncionarioHorista obj2 = new  
                                FuncionarioHorista();  
  
        obj2.setNome("Hulk");  
        obj2.setNumeroMatricula(456);  
        obj2.setValorHora(90.0);  
        obj2.setHorasTrabalhadas(100);  
  
        obj2.imprimirDados();  
    }  
}
```



- 456
- Hulk
- 90.0
- 100

→ obj2





Classe Abstrata - Resumo

- Atributos e construtores não podem ser abstratos.
- Uma classe que contém um ou mais métodos abstratos deve ser declarada como abstrata.
- Toda subclasse concreta de uma superclasse abstrata deve implementar os métodos abstratos da superclasse.
- Se uma classe estende outra que tem método abstrato e não o implementa, então ela também deve ser declarada como abstrata.



Referências

- Deitel, P. J.; Deitel, H. M. (2017). Java como programar. 10a edição. São Paulo: Pearson Prentice Hall.
- Barnes, D. J. (2009). Programação orientada a objetos com Java: uma introdução prática usando o BlueJ (4. ed.). São Paulo, SP: Prentice Hall.
- Boratti, I. C. (2007). Programação orientada a objetos em Java. Florianópolis, SC: Visual Books.