

Herança e Polimorfismo

EXERCÍCIO 35

Leia o capítulo 7 da apostila e resolva os exercícios.

EXERCÍCIO 36

Considere, como subclasse da classe `Pessoa` a classe `Fornecedor`. Cada instância da classe `Fornecedor` tem, além dos atributos que caracterizam a classe `Pessoa`, os atributos `creditoMax` (correspondente ao crédito máximo atribuído pelo fornecedor a determinado cliente) e `valorEmDivida` (montante da dívida para com o fornecedor). Implemente na classe `Fornecedor`, além dos usuais métodos `setters/getters` e modificadores, um método `obterSaldo` que devolve a diferença entre os valores dos atributos `creditoMax` e `valorEmDivida`.

EXERCÍCIO 37

Depois de implementada a classe `Fornecedor`, crie um programa de teste adequado que lhe permita verificar o funcionamento dos métodos implementados na classe `Fornecedor` e os herdados da classe `Pessoa`.

EXERCÍCIO 38

Considere, como subclasse da classe `Pessoa`, a classe `Empregado`. Considere que cada instância da classe `Empregado` tem, além dos atributos que caracterizam a classe `Pessoa`, os atributos `numeroSecao`, `salarioBase` (vencimento base) e `IR` (porcentagem retida para o Imposto de Renda). Implemente a classe `Empregado` com métodos `setters/getters` e modificadores e um método `calcularSalario` (`salarioBase - IR`). Escreva um programa de teste adequado para esta classe.

EXERCÍCIO 39

Considere, como subclasse da classe `Pessoa` a classe `Cliente`. Considere que cada instância da classe `Cliente` tem, além dos atributos que caracterizam a classe `Pessoa`, os atributos `creditoMax` (correspondente ao crédito máximo concedido ao cliente) e `valorEmDivida`. Implemente na classe `Cliente`, um método `obterSaldo` que devolve a diferença entre os valores dos atributos `creditoMax` e `valorEmDivida`. Escreva um programa de teste adequado para esta classe.

EXERCÍCIO 40

A classe `Funcionário` tem, além dos atributos da classe `Pessoa`, os atributos `numeroSecao`, `salarioBase` (vencimento base) e `IR` (porcentagem retida para o Imposto de Renda). Implemente a classe `Funcionário` com métodos `setters/getters` e modificadores e um método `calcularSalario` (`salarioBase - IR`).

Implemente a classe `Funcionário` de duas maneiras:

- `Funcionário` tem como atributo um objeto `pessoa` (composição de classes);
- `Funcionário` é subclasse de `Pessoa`.

OBS: classe `Pessoa` da lista 5

EXERCÍCIO 41

Implemente a classe `Administrador` como subclasse da classe `Funcionário`. Um administrador tem, além dos atributos da classe `Funcionário`, o atributo `ajudasDeCusto` (ajudas referentes a viagens, estadias, ...). Note que deverá redefinir (reescrever) na classe `Administrador` o método herdado `calcularSalario`. O salário de um administrador é equivalente ao salário de um funcionário usual acrescido das ajudas de custo. Escreva um programa de teste adequado para esta classe.

EXERCÍCIO 42

Crie um programa de teste que lhe permita verificar o funcionamento dos métodos implementados na classe `Administrador` e os herdados da classe `Funcionário`. No programa teste use as duas implementações da classe `Funcionário`.

