Pontifícia Universidade Católica de Minas Gerais
Instituto de Ciências Exatas e Informática
Algoritmos e Estruturas de Dados 1
Professor Cristiano Rodrigues

Lista 16 - Herança e Polimorfismo - Tópicos avançados em orientação a objetos

1. **Exercício - Ponto, Círculo e Esfera (com herança para prática)**

   (a) Defina uma classe `Ponto`, com atributos privados para coordenadas `x` e `y`.

   (b) Implemente construtores (com e sem parâmetros), métodos para definir e obter a posição do ponto, calcular distância entre pontos (com outro objeto ou coordenadas).

   (c) Defina a classe `Circulo` como derivada de `Ponto`, adicionando um atributo `raio` e métodos para calcular área e circunferência.

   (d) Defina a classe `Roda` como derivada de `Circulo`, adicionando atributos como `material`, `aro` e `fabricante`.

   (e) Defina a classe `Esfera` como derivada de `Circulo`, adicionando um método para calcular o volume:

   $$\text{Volume} = \frac{4}{3}\pi r^3$$

   (f) Crie um programa principal para instanciar um ponto, um círculo, uma roda e uma esfera, testando os métodos definidos.

2. **Gerenciamento de Dados de Carros com Structs (revisão)**

   a) Crie uma struct `Data` com `dia`, `mes`, `ano`.

   b) Crie uma struct `Carro` com modelo, marca, ano de fabricação, preço e uma data de venda (do tipo `Data`).

   c) Implemente a função `carroMaisCaro` que recebe um vetor de carros e mostra o preço e a data do carro mais caro.

3. **Herança - Cálculo de Salário de Vendedores**

   a) Implemente a classe `Empregado` com nome (`string`), salário base (`double`) e imposto (`double`). Inclua construtores, getters e setters.

   b) Implemente a classe `Vendedor`, derivada de `Empregado`, com atributos adicionais `valorVendas` e `comissao`.

   c) Implemente o método `calcularSalario()` em `Vendedor`, que retorna o salário líquido considerando:

   $$salarioLiquido = (salarioBase + (comissao \times valorVendas)) \times (1 - imposto)$$

   d) No programa principal, crie dois vendedores e exiba seus salários líquidos.

4. **Polimorfismo - Formas Geométricas**

   a) Crie uma classe base `Forma`, com um método virtual `calcularArea()`.

   b) Crie duas classes derivadas: `Retangulo` e `Triangulo`, que sobrescrevem o método `calcularArea()`.

c) O `Retangulo` deve receber base e altura, e retornar `base * altura`.

d) O `Triangulo` deve retornar `(base * altura)/2`.

e) No programa principal, declare um vetor de ponteiros para `Forma`, instancie um retângulo e um triângulo, e use o polimorfismo para calcular as áreas.

5. **Hierarquia de Veículos com Múltipla Herança**

a) Crie a classe `Veiculo` com os atributos `peso`, `velocidadeMaxima` e `preco`.

b) Crie a classe `Motor` com os atributos `numCilindros` e `potencia`.

c) Crie a classe `CarroPasseio`, derivada de `Veiculo` e `Motor`, com os atributos adicionais `modelo` e `cor`.

d) Crie a classe `Caminhao`, também derivada de `Veiculo` e `Motor`, com os atributos adicionais `toneladas`, `alturaMaxima` e `comprimento`.

e) Para cada classe, implemente construtores, getters, setters e um método `print()` para exibir os dados.

f) No programa principal, instancie objetos de `Veiculo`, `Motor`, `CarroPasseio` e `Caminhao`. Configure seus atributos e imprima os dados.

6. **Exercício Final – Sistema de Gestão de Funcionários**

Crie uma hierarquia de classes para representar diferentes tipos de funcionários em uma empresa. A classe base deve ser `Funcionario`, e as classes derivadas devem ser `Gerente`, `Desenvolvedor` e `Estagiario`.

a) A classe `Funcionario` deve conter:

- `nome` (string), `salarioBase` (double);
- Um método virtual `calcularSalario()` que retorna o salário total do funcionário;
- Um método `exibirDados()` que mostra nome e salário final.

Obs.: o método `calcularSalario()` deve ser declarado como `virtual` na classe `Funcionario`, permitindo que seja sobrescrito (override) nas classes derivadas, conforme o comportamento específico de cada tipo de funcionário.

b) A classe `Gerente` deve herdar de `Funcionario` e incluir:

- bônus fixo de R$ 2000,00 no salário;
- sobrescrever o método `calcularSalario()`.

c) A classe `Desenvolvedor` deve herdar de `Funcionario` e incluir:

- número de projetos concluídos (int);
- um bônus de R$ 500,00 por projeto no salário;
- sobrescrever o método `calcularSalario()`.

d) A classe `Estagiario` deve herdar de `Funcionario` e incluir:

- percentual de bolsa (float, entre 0.0 e 1.0);
- o salário é o `salarioBase` multiplicado por esse percentual;
- sobrescrever o método `calcularSalario()`.

e) No programa principal:

- Crie um vetor de ponteiros para `Funcionario`;

- Instancie pelo menos um objeto de cada tipo (Gerente, Desenvolvedor e Estagiario);
- Use polimorfismo para calcular e exibir os salários de todos, por meio do método `exibirDados()`.