Programação orientada a objetos

Prof. Rommel Dias

Exercício 1 *Crie uma classe chamada Conta que tem como atributos o número da agência, o número da conta corrente e o saldo presente na conta. Além disso:*

- *faça um método que calcule a bonificacação do correntista. A bonificação corresponde a 10% do saldo;*
- *declare como privado todos os atributos da classe;*
- *implemente o construtor da classe;*
- *encapsule todo os atributos.*

Exercício 2 *Adicionar no exercício anterior o atributo "cpf" do titular da conta. Fazer uso do this.*

Exercício 3 *Crie uma classe chamada Invoice que possa ser utilizada por uma loja de suprimentos de informática para representar uma fatura de um item vendido na loja. Uma fatura deve incluir as seguintes informações como atributos: o identificador (String) do item faturado, a quantidade comprada do item e o preço unitário do item.*

- *a inicialização de um objeto do tipo Invoice deve estar de acordo com as seguintes regras: Se a quantidade não for positiva, ela deve ser configurada como 0. Se o preço por item não for positivo ele deve ser configurado como 0.0. Utilize um construtor para isso;*
- *forneça um método que calcula o valor da fatura e depois retorna o valor como um double;*
- *implemente um método para mostrar as informações (atributos) da classe. Em seguida, declare e inicialize um vetor com informações para n Invoice, onde n é um dado de entrada. Mostre as informações de todos os objetos do tipo Invoice cadastrados.*

Exercício 4 *Refaça o Exercício 1 de tal forma que o saldo possa ser modificado.*

Exercício 5 *Crie uma classe para representar uma pessoa, com os atributos privados de nome, ano de nascimento e altura. Crie os métodos públicos necessários para sets e gets e também um método para mostrar todos dados de uma pessoa. Crie um método para calcular a idade da pessoa.*

Exercício 6 *Crie uma classe denominada Elevador para armazenar as informações de um elevador dentro de um prédio. A classe deve armazenar o andar atual (térreo = 0), total de andares no prédio, capacidade do elevador (número máximo de pessoas) e quantas pessoas estão presentes nele. A classe deve também disponibilizar os seguintes métodos:*

- *inicializa: que deve receber como parâmetros a capacidade do elevador e o total de andares no prédio (os elevadores sempre começam no térreo e vazio);*
- *entra: para acrescentar uma pessoa no elevador (só deve acrescentar se ainda houver espaço;*
- *sai: para remover uma pessoa do elevador (só deve remover se houver alguém dentro dele);*
- *sobe: para subir um andar (não deve subir se já estiver no último andar);*
- *desce: para descer um andar (não deve descer se já estiver no térreo);*
- *encapsular todos os atributos da classe (criar os métodos set e get).*