Lista 1 - POO

Prof. Dr. Alexandre Garcia de Oliveira

March 7, 2024

1 Introdução, construtores e enumerações

Exercício 1. Dada a classe Pessoa que possui os atributos: `String nome`, `String sexo`, `int idade`, `boolean vegetariana`. Faça agora uma classe Churrasco que possua: Atributos: `qtdCarne(double)`; Método: `verificarConsumo()`: Recebe via parâmetro uma Pessoa e com isto define a consumação média de carne (quantidade de carne consumida) pessoas de 0 a 3 anos não consomem, vegetarianos também não. Pessoas de 4 a 12 anos consomem 1 kilo de carne e de 13 anos em diante 2 kilos de carne.

Exercício 2. Implemente uma classe Lâmpada com os seguintes componentes e faça um teste ao final: Atributos: `estado(Enum)` Métodos:

- `click()`: ao chamar este método a lâmpada é colocada no estado "apagada" caso esteja "acesa" e é colocado no estado "acesa" caso esteja "apagada".
- `qtdAcendimentos()`: retorna quantas vezes a lâmpada foi acesa (DICA: este método deve ser chamada no método acima).
- `checaEstado()`: retorna o estado atual da Lâmpada.

Exercício 3. Implemente uma classe chamada Complexo para representar números imaginários e esta deve possuir:

- Atributos: dois doubles a (Parte real) e b (Parte imaginária).
- Métodos:
  1. Construtor;
  2. soma(): recebe via parâmetro outro número complexo (objeto desta classe) e efetua sua soma, ou seja, parte real será somada com parte real e parte imaginária com parte imaginária.
  3. multiplica(): recebe via parâmetro outro complexo (objeto desta classe) e efetue a fórmula $(a + bi) \cdot (c + di) = (ac - bd) + (ad + bc)i$
  4. toString(): Mostra uma string na tela com os atributos a e b na notação Complexa $a + bi$;

5. modulo(): retorna o módulo do número complexo que é dado por $\sqrt{a^2 + b^2}$
6. argumentoPrincipal(): retorna o ângulo formado pelo número complexo no plano de Argand-Gauss, que é dado pela fórmula $\theta = \tan^{-1}\left(\frac{b}{a}\right)$

Exercício 4. Implemente a classe Cliente que possua os atributos nome, saldo, e limite. Esta deve possuir também os métodos sacar(), depositar(), checarSaldo(), e obterNome(). Sabe-se que só é possível sacar se o saldo+limite forem superiores à quantia. Os métodos sacar() e depositar() necessitam de parâmetros. O método checarSaldo deve retornar o valor saldo+limite. O método obterNome() deve retornar o nome do Cliente.

Exercício 5. Implemente uma classe que modele um triângulo equilátero (lados iguais). Atributos: lado, perímetro, área. Métodos: calcArea(), calcPerímetro e seus gets. O lado deverá ser o único atributo inicializado via construtor. Fórmulas: Área = $\frac{lado \times \sqrt{3}}{2}$, Perímetro = $3 \times lado$.

Exercício 6. Implemente uma classe que modele um jogo de adivinhação de números de 0 a 99. Atributos: um número inteiro sorteado. Métodos: sortear(), advinhar(). OBS: O objeto para gerar número aleatórios no Java é o Random; você deve instanciá-lo e chamar seu método nextInt() que deve possuir um argumento inteiro, no caso aqui, 100.

Exercício 7. Implemente a classe Eq2Grau que possua: Atributos: a, b e c (doubles); Métodos: delta(): retorna o delta da equação; raiz1(): retorna a primeira raiz se $\Delta \geq 0$, se não, retorna NaN; raiz2(): retorna a segunda raiz se $\Delta \geq 0$, se não, retorna NaN.

2 Modificador static

Exercício 8. Implemente a classe Porta que possua: Atributos: isOpen(boolean), numAberturas(int); Métodos: abrir(): abre a porta e conta 1 na contagem de aberturas; fechar(): fecha a porta. OBS: O atributo numAberturas deve contar o total de aberturas de todas as portas possíveis.

3 Relação entre classes

Exercício 9. Usando o Exercício 4, implemente a classe Transferencia que possuí o método transferir() que recebe via parâmetro dois Clientes c1 e c2 (ver exercício acima) e a quantia (necessita uma verificação de saldo). Deve ser tirado da conta de c2 e colocado na conta de c1. Exiba também uma mensagem de conclusão de transferência explicitando os nomes dos envolvidos.

Exercício 10. Usando o exercício 8, faça uma classe `Casa` com que represente uma casa. A casa tem uma cor, uma porta de entrada e pode ter até três portas

(considere que todas as portas podem ser abertas ou fechadas independentemente). Implemente os métodos para abrir e fechar as portas, e um método que retorne quantas portas estão abertas.

Exercício 11. Usando o exercício anterior, implemente a classe `Edificio` que possui vários apartamentos. Suponhamos, por simplicidade, que cada edifício possui apenas 3 apartamentos. Crie uma classe `Apartamento` que possui um número identificador. A classe `Edificio` deve ser capaz de adicionar apartamentos, listar todos os apartamentos e buscar um apartamento pelo seu identificador.

Exercício 12. Implemente a classe Cliente que contenha os atributos: nome, cpf (Strings) e telefone (Telefone). E que contenha os métodos: mostrarDados() e adicionarTelefone() o primeiro deve mostrar todos os dados do cliente incluindo o telefone e o último deve associar um novo telefone ao cliente. Implemente a classe Telefone que possua os atributos: ddd e número (Strings) e os métodos: obterNumero() e obterDDD().