Lista de Exercícios

Membros de Classe com static

1. **Contador de Livros:** Implemente o contador estático na classe Livro conforme visto em aula. Crie o atributo private static int contadorDeLivros, incremente-o no construtor e crie o método public static int getContadorDeLivros() para acessá-lo.
2. **Imposto Compartilhado:** Crie uma classe Produto. Todo Produto tem um private double preco. A loja tem uma taxa de imposto que é a **mesma para todos os produtos**. Crie um atributo private static double taxaImposto = 0.05; (representando 5%). Crie um método de instância public double getPrecoComImposto() que retorna this.preco * (1 + taxaImposto).
3. **Alterando a Regra para Todos:** Na classe Produto do exercício anterior, crie um método estático public static void setTaxaImposto(double novaTaxa). No main, crie dois produtos, exiba seus preços com imposto. Em seguida, chame Produto.setTaxaImposto(0.10);. Exiba novamente os preços com imposto dos dois produtos para provar que a alteração da variável estática afetou ambos.
4. **A Analogia (Conceitual):** Qual a diferença de memória entre o atributo preco e o atributo taxaImposto da sua classe Produto? Use a analogia da "República e o Monumento" para explicar.
5. **Regras do static (Conceitual):** É possível, dentro de um método static como setTaxaImposto, acessar um atributo de instância como this.preco? Explique com suas palavras por que sim ou por que não.

Constantes e Classes Utilitárias

6. **Constantes do Sistema:** Crie uma classe ConfiguracaoSistema. Dentro dela, declare duas constantes:
    - public static final int NUMERO_MAXIMO_LOGIN_TENTATIVAS = 5;
    - public static final String URL_BLOG_DA_EMPRESA = "https://meublog.com.br";

No main, acesse e imprima esses valores diretamente da classe.

7. **Caixa de Ferramentas de Validação:** Crie uma classe utilitária ValidacaoUtil com um construtor privado. Adicione a ela um método estático public static boolean isEmailValido(String email). A validação pode ser simples: verifique se o email contém o caractere @ e ..
8. **Usando a Ferramenta:** Crie uma classe Usuario com um setter setEmail(String email). Dentro deste setter, antes de atribuir o valor, use a sua ferramenta ValidacaoUtil.isEmailValido(email) para verificar se o email é válido.
9. **Caixa de Ferramentas de Arrays:** Crie uma classe ArrayUtil com um construtor privado. Adicione um método estático public static int encontrarMaiorNumero(int[] numeros) que recebe um array de inteiros e retorna o maior valor contido nele.
10. **Decisão de Design (Conceitual):** Você precisa criar uma funcionalidade para calcular juros compostos. O método seria calcularJurosCompostos(double principal, double taxa, int periodos). Você criaria uma classe CalculadoraDeJuros para ser instanciada (new CalculadoraDeJuros()) ou a implementaria como um método estático em uma classe FinanceiroUtil? Justifique sua decisão de design.