Curso: Tecnólogo em Análise e Desenvolvimento de Sistemas
Disciplina: Programação Orientada a Objetos
Prof: Francisco do Nascimento

LISTA DE EXERCÍCIOS 03

PARTE A: ENCAPSULAMENTO

1. Considere o código da classe Produto:

```
public class Produto {
	private String nome;

	public String getNome() {
		return nome;
	}
	public void setNome(String nome) {
		this.nome = nome;
	}
	public static void alterarNome(Produto prod, String nome) {
		prod = new Produto();
		prod.setNome(nome);
	}
	public static void main(String[] args) {
		Produto p1 = new Produto();
		p1.setNome("Biscoito Treloso");
		Produto p2 = new Produto();
		p2.setNome("Doritos");
		alterarNome(p1, "Biscoito Passatempo");
		System.out.println(p1.getNome());
		System.out.println(p2.getNome());
	}
}
```

Qual seria a saída esperada da execução da classe Produto?

a) Erro de compilação
b) Biscoito Passatempo
   Doritos
c) Biscoito Treloso
   Doritos
d) Biscoito Passatempo
   Biscoito Passatempo
e) Biscoito Treloso
   Biscoito Treloso

2. Analise o código da classe Cliente e escreva a saída da execução:

```
public class Cliente {
	private static int ULTIMO_CODIGO = 1;
	private int codigo;
	private String nome;

	public Cliente() {
		codigo = ULTIMO_CODIGO++;
	}
	public Cliente(String nome) {
		this();
		this.nome = nome;
	}
	public Cliente(Integer c, String n) {
		this.codigo = c;
		this.nome = n;
	}
	public int getCodigo() {
		return codigo;
	}
	public void setCodigo(int codigo) {
		this.codigo = codigo;
	}
	public String getNome() {
		return nome;
	}
	public void setNome(String nome) {
		this.nome = nome;
	}
	public static void main(String[] args) {

		Cliente c1 = new Cliente();
		Cliente c2 = new Cliente("Poliana");
		Cliente c3 = new Cliente(45, "Poliana");
		Cliente c4 = new Cliente();
		c4.setNome("Carlos");
		c4.setCodigo(30);
		c4 = new Cliente("Nelson");

		System.out.println(c1.getCodigo());
		System.out.println(c2.getCodigo());
		System.out.println(c3.getCodigo());
		System.out.println(c4.getCodigo());

	}
}
```

PARTE B: java.lang.String, java.lang.Math

3. Criar uma classe Funcionario com as seguintes características:
   - atributos: cpf, nome, salario
   - construtor: vazio
   - métodos:
     - getters/setters para cada atributo
     - obterQuantidadeLetrasNome(): retornar a quantidade de letras do nome da pessoa
     - obterNomeAbreviado(): retornar apenas o primeiro e o último nome da pessoa
     - obterIniciais(): retornar uma string contendo apenas as letras iniciais do nome da pessoa
     - verificarCPFFormatado(): retornar true se o CPF estiver exatamente no formato 999.999.999-99, onde o 9 pode ser qualquer dígito de 0-9
     - formatarCPF(): retornar o CPF do funcionário aplicando a formatação 999.999.999-99, caso ele não esteja formatado.
     - obterNumeroSalariosMinimos(): retornar um inteiro aproximado para cima que representa a quantidade de salários mínimos que o funcionário recebe.
     - adicionarAumentoSalario(percentual): alterar o salário do funcionário aplicando o percentual informado.

4. Criar uma classe TesteFuncionario com o método main e implementar as seguintes ações:
   - Instanciar dois funcionários:

CPF	NOME	SALÁRIO
43500344322	Júlio dos Santos	R$ 3.450,00
304.304.223-33	Adriana Milena da Paz e Souza	R$ 8.550,40

   - Imprimir o nome de cada funcionário: caso o nome tenha mais de 20 caracteres, exibir o nome abreviado, caso contrário, exibir o nome inteiro.
   - Imprimir o CPF formatado de cada funcionário
   - Realizar um aumento de 10% no salário de cada funcionário
   - Imprimir o número de salários mínimos que representa o quanto cada funcionário recebe.

PARTE C: java.time.LocalDate

5. Criar uma classe Boleto com as seguintes características:
   - atributos: fornecedor, valor, data de vencimento
   - construtores: todos os atributos
   - métodos:
     - getters/setters para cada atributo
     - verificarEmDia: retornará true se o boleto está em dia, false se estiver atrasado
     - calcularMulta: retornar o valor da multa, considerando que seja 1% por dia de atraso
     - prorrogarVencimento: alterar o vencimento para mais 5 dias
     - calcularDesconto: retornar o valor do desconto, considerando que seja 0.5% por dia de adiantamento
     - calcularValorFinal: retornar o valor final do boleto, adicionando o valor da multa ou do desconto, se for o caso.
     - equals: considerar dois boletos iguais se tiverem o mesmo fornecedor e valor

6. Criar uma classe TesteBoleto com o método main e implementar as seguintes ações:
   - Instanciar 3 objetos da classe Boleto:
     - Boleto 1: Fornecedor "Ambev", vencimento 15/04/2023, valor R$ 1.560,00
     - Boleto 2: Fornecedor "Coca-Cola", vencimento 10/04/2023, valor R$ 2.450,00
     - Boleto 3: Fornecedor "Ambev", vencimento 04/04/2023,valor R$ 1.560,00
   - Imprimir o valor final de cada boleto
   - Prorrogar o vencimento do Boleto 3
   - Imprimir a mensagem "Boletos iguais", caso o boleto 1 seja igual ao boleto 3.
   - Imprimir a soma dos três boletos se forem pagos no dia 30/04/2023.