LISTA DE EXERCÍCIOS PARA ORIENTAÇÃO A OBJETOS

ALUNO	Silvya Daniele da Silva Oliveira	MATRÍCULA	04155813
ALUNO	Matheus Reis	MATRÍCULA	04163098

1. Classe Fatura. Crie uma classe chamada Fatura para uma loja de suprimentos de informática com os seguintes atributos:

- codigoProduto (String)
- descricaoProduto (String)
- quantidadeComprada (int)
- precoPorItem (double)

A classe deve possuir:

- Um construtor que inicializa as variáveis de instância.
- Métodos get e set para cada variável.
- Um método getTotalFatura(), que calcula o valor total da fatura e retorna o valor como um double. Se a quantidade ou o preço não forem positivos, devem ser configurados como 0.

Crie uma classe de teste chamada **FaturaTeste** em um arquivo separado, que instancie um objeto da classe Fatura e demonstre suas funcionalidades, incluindo a exibição do valor total da fatura.

2. Classe Empregado. Crie uma classe chamada Empregado com os seguintes atributos:

- nome (String)
- sobrenome (String)
- salarioMensal (double)

A classe deve possuir:

- Um construtor que inicializa as variáveis de instância.
- Métodos get e set para cada variável.
- Um método que calcula e retorna o salário anual.

Crie uma classe de teste chamada EmpregadoTeste, que instancie dois objetos Empregado, exiba o salário anual de ambos, aplique um aumento de 10% no salário e exiba o novo salário anual.

3. Classe Data. Crie uma classe chamada Data com os seguintes atributos:

- mes (int)
- dia (int)
- ano (int)

A classe deve possuir:

- Métodos get e set para cada variável.
- Um construtor que inicializa as variáveis de instância assumindo que os valores fornecidos são válidos.
- Um método displayData() que exibe a data no formato "dia/mês/ano".

Crie uma classe de teste chamada DataTeste que instancie um objeto Data e demonstre suas funcionalidades, exibindo a data formatada.

4. Classe InteiroSet. Crie uma classe chamada InteiroSet que armazena inteiros no intervalo de 0 a 100 utilizando um array de booleans. Cada índice do array representa um número; o valor true indica que o número está presente no conjunto, e false indica que não está. A classe deve possuir os seguintes métodos:

- union(InteiroSet outroConjunto): Retorna um novo conjunto que representa a união dos dois conjuntos.
- intersecao(InteiroSet outroConjunto): Retorna um novo conjunto que representa a interseção dos dois conjuntos.
- insereElemento(int k): Insere o elemento k no conjunto.
- deleteElemento(int m): Remove o elemento m do conjunto.
- toSetString(): Retorna uma string com os elementos do conjunto separados por espaços. Se o conjunto estiver vazio, retorna "-".
- ehIgualTo(InteiroSet outroConjunto): Verifica se dois conjuntos são iguais.

Implemente uma classe de teste para demonstrar a criação de dois conjuntos, a inserção de elementos, a união e a interseção dos conjuntos, e a comparação de igualdade entre eles.

5. Hierarquia de Veículos. Crie uma hierarquia de classes para representar diferentes tipos de veículos. Defina uma classe base chamada Veiculo com os seguintes atributos:

- marca (String)
- modelo (String)
- ano (int)

A classe deve possuir:

- Métodos get e set para cada atributo.
- Um método e() que exibe as informações do veículo.

Crie duas classes derivadas:

- Carro: Adicione o atributo numeroDePortas (int) e sobrescreva o método exibirDetalhes() para incluir o número de portas.
- Moto: Adicione o atributo tipoDeGuidon (String) e sobrescreva o método exibirDetalhes() para incluir o tipo de guidão.

Instancie objetos das classes Veiculo, Carro e Moto, e utilize o método exibirDetalhes() para demonstrar a herança e a sobrescrita de métodos.

6. Sistema de Funcionários. Crie uma classe base chamada Funcionario com os seguintes atributos:

- nome (String)
- salarioBase (double)

A classe deve possuir:

- Métodos get e set para cada atributo.
- Um método calcularSalario() que retorna o valor do salário base.

Crie duas classes derivadas:

- Gerente: Adicione o atributo bonus (double) e sobrescreva o método calcularSalario() para adicionar o bônus ao salário base.
- Assistente: Mantenha o método calcularSalario() sem alterações.

Instancie objetos das classes Gerente e Assistente, e utilize o método calcularSalario() para calcular e exibir o salário de cada funcionário.

7. Animais e Sons. Crie uma classe base chamada Animal com os seguintes atributos:

- nome (String)
- idade (int)

A classe deve possuir:

- Métodos get e set para cada atributo.
- Um método abstrato ou padrão emitirSom() que retorna uma mensagem genérica como "O animal faz um som".

Crie duas classes derivadas:

- Cachorro: Sobrescreva o método emitirSom() para retornar "O cachorro late".
- Gato: Sobrescreva o método emitirSom() para retornar "O gato mia".

Instancie objetos das classes Cachorro e Gato, e chame o método emitirSom() para demonstrar a sobrescrita de métodos.

8. Produtos e Descontos. Crie uma classe base chamada Produto com os seguintes atributos:

- nome (String)
- preco (double)

A classe deve possuir:

- Métodos get e set para cada atributo.
- Um método calcularPrecoComDesconto() que retorna o preço do produto (sem desconto, por padrão).

Crie duas classes derivadas:

- Eletronico: Adicione o atributo garantia (int) e sobrescreva o método calcularPrecoComDesconto() para aplicar um desconto de 10% no preço.
- Alimento: Adicione o atributo dataDeValidade (String) e mantenha o método calcularPrecoComDesconto() sem alterações.

Instancie objetos das classes Eletronico e Alimento, e utilize o método calcularPrecoComDesconto() para demonstrar a aplicação de desconto no preço do produto.