Exercícios de Fixação – Java Introdutório

Instruções para os exercícios de 2 à 5:

1. Crie um único projeto Java para a implementação dos exercícios;
2. Organize cada exercícios e suas devidas implementações em pacotes (`packages`) diferentes no projeto Java; e
3. Ao finalizar o exercício e antes de postá-lo para entrega, siga os passos de Emportação do Projeto como *Archive File* descritos no link https://goo.gl/sQMuY4. Ao exportar o projeto, remova as pastas **bin** e **.settings**.

Exercício 1. Descreva em no máximo 200 palavras o que é um automóvel e o que ele faz. Liste os substantivos e verbos separadamente. Cada substantivo corresponde a um objeto que precisará ser construído para implementar um sistema, nesse caso, um carro. Selecione 5 dos objetos que você listou e, para cada um, liste vários atributos e comportamentos. Descreva brevemente como esses objetos interagem entre si e com outros objetos na sua descrição. Estes passos que você seguiu são típicos do projeto orientado a objetos.

Exercício 2. Crie uma classe chamada `Invoice` que possa ser utilizado 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 número do item faturado;
- a descrição do item;
- a quantidade comprada do item; e
- o preço unitário do item.

Sua classe deve ter um construtor que inicialize os quatro atributos. 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`. Forneça um método set e um método `get()` para cada variável de instância. Além disso, forneça um método chamado `getInvoiceAmount()` que calcula o valor da fatura (isso é, multiplica a quantidade pelo preço por item) e depois retorna o valor como um `double`. Escreva um aplicativo de teste que demonstra as capacidades da classe `Invoice`.

Exercício 3. A fim de representar empregados em uma firma, crie uma classe chamada `Empregado` que inclui as três informações a seguir como atributos:

- um primeiro nome;
- um sobrenome; e
- um salário mensal.

Sua classe deve ter um construtor que inicializa os três atributos. Forneça um método `set()` e `get()` para cada atributo. Se o salário mensal não for positivo, configure-o como `0.0`. Escreva

um aplicativo de teste que demonstra as capacidades da classe. Crie duas instâncias da classe e exiba o salário anual de cada instância.
Então dê a cada empregado um aumento de 10% e exiba novamente o salário anual de cada empregado.

Exercício 4. Cria uma classe chamada `Complex` para representar números complexos e escreva um programa para testá-la:

1. Escolha uma representação para os números complexos, usando a **forma retangular** ou a **forma polar**;
2. Forneça três construtores que permitam que objetos dessa classe sejam inicializados ao serem alocados na memória:
    - um construtor sem parâmetros que inicializa o objeto como zero;
    - um construtor com um parâmetro representando a parte real e a parte imaginária será zero; e
    - um construtor com dois parâmetros representando as partes real e imaginária respectivamente.
3. Defina operações para obter a parte real, a parte imaginária, o módulo (valor absoluto) e o ângulo de um número complexo;
4. Forneça a operação para determinar o inverso aditivo de um número complexo;
5. Forneça as operações aritméticas básicas com números complexos: adição, subtração, **multiplicação** e divisão;
6. Defina a operação `toString()` para converter um número complexo em String. Utilize o formato `(a,b)`, onde a é a parte real e b é a parte imaginária; e
7. Escreva um aplicativo de teste que demonstra as capacidades da classe `Complex`, executando todas as operações implementadas.

Exercício 5. Crie uma classe para representar datas:

1. Represente uma data usando três atributos: o dia, o mês, e o ano;
2. Sua classe deve ter um construtor que inicializa os três atributos e verifica a validade dos valores fornecidos;
3. Forneça um construtor sem parâmetros que inicializa a data com a data atual fornecida pelo Sistema Operacional (Usar: https://goo.gl/LMRXik);
4. Forneça um método `set()` um `get()` para cada atributo;
5. Forneça o método `toString()` para retornar uma representação da ata como string. Considere que a data deve ser formatada mostrando o dia, o mês e o ano separados por barra (/);
6. Forneça uma operação para avançar uma data para o dia seguinte;
7. Escreva um aplicativo de teste que demonstra as capacidades da classe; e

8. Garanta que uma instância desta classe sempre esteja em um estado consistente.