Lista de exercícios POO

Prof. Maurício Acconcia Dias

Tópicos Avançados em OO

Exercício 1 –

Um jardim zoológico definiu a seguinte interface que estende a interface Animal:

```
public interface AnimalOrcamento extends Animal {
    public ItemOrcamentoComplexo orcamentoGastosAnimal();
}
```

O método `orcamentoGastosAnimal` retorna o orçamento para gastos de um animal no zoológico.

O zoológico deseja saber quais de seus animais têm a "vacina W" prevista no seu orçamento.

Escreva um método que receba como parâmetro um vetor de objetos que implementam a interface `AnimalOrcamento` representando todos os animais do zoológico e seus respectivos orçamentos.

O método deve retornar um outro vetor de objetos que implementam a interface `AnimalOrcamento` apenas com aqueles animais que possuem um subitem com histórico "vacina W" prevista no seu orçamento.

Nesta questão basta implementar o método, não é necessária a especificação da classe.

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 –

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.
4. Forneça um método *set* um *get* para cada atributo.
5. Forneça o método `toString` para retornar uma representação da data 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.

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