Lista de exercícios 4

Programação Orientada à Objetos

Todos exercícios deverão ser entregues no Moodle, em um único arquivo comprimido (.rar de preferência). Os códigos deverão seguir um estilo de código padrão e deverão estar comentados.

1) Modifique o código feito na lista 3 da seguinte forma:

- crie uma classe Cliente, onde serão guardados o nome de cada cliente. Isso implicará em remover os atributos nomeCliente e cpfCliente de ContaBancaria e substituí-los por um objeto da classe Cliente (suponhamos que uma conta pertença a apenas um cliente).
- Crie duas classes PessoaFisica e PessoaJuridica que herdam de Cliente (ou seja, um cliente pode ser tanto uma pessoa física quanto uma empresa). Apenas pessoas físicas têm CPF enquanto pessoas jurídicas têm CNPJ.
- O atributos CPF e CNPJ não poderão ser associados a qualquer valor. Suponha que para atribuir qualquer valor à esses atributos seja necessário passar por uma validação (não é necessário implementar essa validação).
- Crie uma classe Cartao, que tem um número e uma conta associada (objetos da classe ContaBancaria têm uma lista de cartões associados). Crie um método adicionarCartão na classe ContaBancaria, que adiciona um cartão à conta (especifique os atributos do cartão).
- Crie duas classes: CartaoCredito e CartaoDebito. Cartões de crédito tem uma propriedade limite que indica o gasto máximo nesse cartão, uma taxa de juros e um atributo valorGasto.
- Modifique o método saque para que ele também receba como parâmetro o Cartão utilizado para o saque (simulando um caixa eletrônico). Caso o cartão for do tipo crédito, o saque não poderá ser feito, senão o saque ocorre normalmente.
- Crie um método paga(Cartao cartaoPgto, ContaBancaria contaPgto, double valor). Se o cartão utilizado for o cartão de débito, esse método reduz da conta pagante valor e deposita na conta recebedora valor. Se o cartão for de crédito, ele adiciona ao cartão esse valor (caso esteja no limite, senão avisa que não pode realizar a operação) e deposita na conta recebedora o valor.
- Crie um método pagaCartoes() que varre os cartões de crédito e reduz seus gastos do saldo.
- Na main, escreva um código que teste todas essas funcionalidades.

2) Suponha que exista uma classe derivada B que herda de uma classe base A. Se A tem um atributo att1 privado, B herdará esse atributo, mas não poderá acessá-lo de dentro da classe B.

- A tem getAtt1 e setAtt1. Nesse caso, se B precisar usar o atributo att1, é melhor usar getAtt1 e setAtt1 ou modificar a visibilidade de att1 para protected e usar att1 diretamente em B? Justifique sua linha de raciocínio.

- A não tem getAtt1 e setAtt1. Nesse caso, se B precisar usar o atributo att1, é melhor criar um getAtt1 e setAtt1 em A ou modificar a visibilidade de att1 para protected e usar att1 diretamente em B? Justifique sua linha de raciocínio.
- Pense em um exemplo onde attr1 deva ter visibilidade protected e não ser private com getters e setters.