Ficha de exercícios 3

Ex. 1: Crie um novo projeto e codifique a classe ContaBancária com os seguintes atributos e métodos:

a) Uma variável de instância de acesso privado, titular, com tipo String.
b) Uma variável de instância de acesso privado, saldo, com tipo double, e com valor 0.0 por defeito.
c) Uma variável de instância de acesso privado, dataAbertura, com tipo Date, e com valor por defeito a data de hoje.
d) Um construtor de instância da classe que recebe por parâmetro o nome do titular e cria uma conta bancária para esse titular, na data de hoje, e com saldo zero.
e) Os métodos getters e setters para o saldo. E, apenas o método getter para a data de abertura e para o titular.
f) Um método getInformaçãoConta() que retorna uma String com o titular, o saldo da conta com duas casas decimais e a data de abertura da conta no formato DD/MM/YYYY. Ex.: Titular: José Fonseca Saldo: €321.50 Data Abertura: 21/2/2017.
g) Um método depositar(…) que recebe o valor a depositar na conta (double) e atualiza o respetivo saldo.
h) Um método levantar(…) que recebe o valor a levantar da conta (double) e atualiza o respetivo saldo. Não devem ser permitidos levantamentos que coloquem o saldo negativo.

Escreva a função main() para testar a classe ContaBancária. Crie uma conta, efetue depósitos e levantamentos e imprima no ecrã a informação da conta sempre que atualiza o saldo.

Ex. 2: Codifique a classe Banco com os seguintes atributos e métodos:

a) Um nome e um array de contas de objetos do tipo ContaBancária (ver exercício anterior).
b) Um construtor de instâncias da classe, que recebe por parâmetro o nome do banco e que inicializa o array para cem itens.
c) Os métodos getters e setters para o array contas.
d) O método criaConta(…) que recebe uma contaBancária por parâmetro e acrescenta ao array.
e) O método getConta(…) que recebe o titular por parâmetro e devolve a conta desse titular.

Escreva a função main() para testar a classe Banco. Crie uma conta, efetue depósitos e levantamentos e imprima no ecrã a informação da conta sempre que atualiza o saldo.

Ex. 3: Crie uma classe Casa, a qual deverá ter os seguintes atributos e métodos:

a) Um atributo privado de tipo String, morada, para a morada da casa.
b) Dois atributos de acesso privado, precoCusto e precoVenda, com tipo double, para conterem respetivamente o preço de custo e o preço de venda da casa.
c) Um construtor de instâncias da classe, sem argumentos, que cria uma casa.
d) Um construtor de instâncias da classe que recebe por parâmetro a morada da casa e cria uma casa com essa morada.
e) Os métodos getters e setters dos três atributos definidos anteriormente.

f) Um método de instância getLucro() que retorna o lucro da casa (diferença entre o preço de venda e o preço de custo).
g) Um método de instância getPercentMargemLucro() que retorna a percentagem da margem de lucro (a razão entre o preço de venda e o preço de custo).

Escreva a função main() para testar a classe Casa. Crie três casas com diferentes valores dos seus atributos.

Ex. 4: Novamente na classe Banco, adicione uma lista de casas. Adicione uma função para adicionar uma nova casa. Adicione uma função que recebe a morada de uma casa e remove essa casa da lista de casas do Banco. Adicione um método de instância getLucroPrevisto, que retorna a margem de lucro previsto da venda de todas as casas.

Ex. 5: Faça a modelação do sistema com um diagrama de classes UML que descreve o sistema. Adicione o seu projeto ao GitHub e crie um ficheiro README. Adicione ao ficheiro README uma pequena descrição do projeto e adicione o diagrama UML (pode encontrar ajuda a escrever o ficheiro README aqui).