




Lista de Exercício 1

(1.16) Escreva um modelo Empregado que represente um empregado de uma empresa qualquer. Considere que os atributos nome, departamento, horasTrabalhadasNoMês e salárioPorHora devam ser representados, e que ao menos as operações mostraDados e calculaSalárioMensal sejam implementadas.

(1.26) Imagine que os empregados de uma empresa tenham dois valores de salário para horas trabalhadas, diferenciados entre horas normais e horas extras. Modifique o modelo Empregado (veja o exercício 1.16) para que dois valores de horas trabalhadas e dois valores de salário-hora sejam usados.

(1.27) Modifique a operação calculaSalárioMensal no modelo Empregado (veja o exercício 1.16) para que todos os empregados do departamento Diretoria tenham 10% de bônus salarial.

Conceitos Básicos de Programação OO: Implementar as classes a seguir:

- a) Classe: ContaCorrente.
 - a. Atributos da classe ContaCorrente:
numeroConta (inteiro)  não pode receber um valor menor ou igual a zero; correntista (string)  não pode receber um valor nulo ou string vazia; saldo (double) 
não pode manter um valor menor que zero;
 - b. Implementar apenas um construtor recebendo valores para os atributos numeroConta e correntista da classe ContaCorrente.
 - c. Utilizar a técnica de encapsulamento apresentada no curso e implementar métodos get e set para todos os atributos da classe (ver observação abaixo).
 - d. OBS: Não implementar o método set para o atributo saldo. O valor do atributo saldo só poderá ser alterado através dos métodos creditar, debitar e transferir detalhados a seguir.
 - e. Implementar o método void creditar(double valor) que deposita um valor na conta corrente. OBS: Verificar se o valor informado é maior que zero.
 - f. Implementar o método void debitar(double valor) que realiza um saque na conta corrente. OBS: Verificar se o valor informado é maior que zero

e se há saldo suficiente para realizar o saque.

- g. Implementar o método `void transferir(double valor, ContaCorrente c2)` que realiza uma transferência de um valor da conta corrente para a conta corrente `c2`. OBS: Verificar se o valor informado é maior que zero, se o objeto `c2` não é nulo e se há saldo suficiente para realizar a transferência.
- b) Implementar uma aplicação chamada `AplicacaoContas` que instancia dois objetos do tipo `ContaCorrente`, a partir de dados informados pelo usuário. Em seguida, o programa deve oferecer um menu para o usuário com as seguintes opções:
 - a. Imprimir dados de uma conta. Para esta opção o usuário deverá informar o número da conta;
 - b. Realizar depósito. Para esta opção o usuário deverá informar o número da conta e o valor para depósito;
 - c. Realizar saque. Para esta opção o usuário deverá informar o número da conta e o valor para saque;
 - d. Realizar transferência. Para esta opção o usuário deverá informar o número da conta origem, o número da conta destino e o valor para transferência;
 - e. OBS: Para as opções de depósito, saque e transferência, o programa deve imprimir na tela uma mensagem indicando se o depósito, o saque ou a transferência foi realizado com sucesso ou não.

Conceitos Básicos de Programação OO: Implementar as classes a seguir:

c) Classe: DVD.

- Atributos da classe DVD:

estado (booleano) receberá o valor true quando o DVD estiver ligado, e false em caso contrário. Defina, na classe DVD, duas constantes booleanas: LIGADO = true e DESLIGADO = false.

operacaoEmExecucao (string) receberá um dos valores pré-definidos (ver tabela abaixo) de acordo com a operação que estiver sendo realizada pelo DVD. OBS: definir constantes para manter os valores da tabela abaixo:

Constante	Valor (string)
PARADO	"" (string vazia)
REPRODUZINDO	"REPRODUZINDO O DVD..."
EM_PAUSA	"REPRODUÇÃO DO DVD EM PAUSA..."
VOLTANDO	"VOLTANDO O DVD..."
AVANÇANDO	"AVANÇANDO O DVD..."

- Implementar apenas o construtor default que atribui o valor DESLIGADO para o atributo estado e o valor PARADO para o atributo operacaoEmExecucao.
- Utilizar a técnica de encapsulamento apresentada no curso e implementar apenas métodos get para todos os atributos.
- OBS: Os métodos set dos atributos estado e operacaoEmExecucao não serão implementados pois seus valores só poderão ser alterados através dos métodos liga, desliga, play, stop, pause, volta e avança detalhados a seguir.
- Implementar o método void liga() que altera o estado do equipamento para LIGADO e atribui o valor PARADO para o atributo operacaoEmExecucao.
- Implementar o método void desliga() que altera o estado do equipamento para DESLIGADO e atribui o valor PARADO para o atributo operacaoEmExecucao.
- Implementar o método void play() que altera o valor do atributo operacaoEmExecucao para REPRODUZINDO. OBS: Este método só pode ser executado se o estado do equipamento for LIGADO.
- Implementar o método void stop() que altera o valor do atributo operacaoEmExecucao para PARADO. OBS: Este método só pode ser executado se o estado do equipamento for LIGADO.
- Implementar o método void pause() que altera o valor do atributo operacaoEmExecucao para EM_PAUSA. OBS: Este método só pode

ser executado se o estado do equipamento for LIGADO.

- Implementar o método void volta() que altera o valor do atributo operacaoEmExecucao para VOLTANDO. OBS: Este método só pode ser executado se o estado do equipamento for LIGADO.
 - Implementar o método void avanca() que altera o valor do atributo operacaoEmExecucao para AVANCANDO. OBS: Este método só pode ser executado se o estado do equipamento for LIGADO.
- d) Implementar uma aplicação chamada AplicacaoDVD que instancia um objeto do tipo DVD e oferece um menu com as opções: liga, desliga, play, stop, pause, volta, avança e sair, permitindo que o usuário possa manipular seu equipamento. OBS: Todas as vezes que o usuário selecionar uma opção do menu, deve-se imprimir na tela o estado e a operação corrente do equipamento.

❖ Implementar a classe Quadrado com as seguintes definições:

- ❖ O atributo lado;

- ❖ Dois construtores, sendo um deles default.

- ❖ O método double area()

- ❖ O método double comprimento()

- ❖ O método void desenha();

❖ Implementar a aplicação AplicaçãoQuadrado que cria um objeto do tipo Quadrado, a partir do lado informado pelo usuário, e imprime o desenho do quadrado, o valor da sua área, comprimento e lado.

❖ Implementar a classe TrianguloRetangulo com as seguintes definições:

- ❖ Os atributos base, altura e hipotenusa;

- ❖ Dois construtores, sendo um deles default;

- ❖ O método double area();

- ❖ O método double comprimento();

❖ Implementar a aplicação AplicacaoTriangulo que cria um objeto do tipo TrianguloRetangulo, a partir de valores informados pelo usuário, e imprime o valor da sua área, comprimento e atributos.