**Exercícios**

**Programação Orientada a Objetos**

1. Criar uma classe Time para armazenar internamente Hora, Minutos e Segundos.

* Não permita atribuição direta e só aceitar valores válidos (0-23 ou 0-59)
* Criar um método setTime para passar a hora, minuto e segundos de uma só vez
* Implementar um método set e um get para cada atributo
* Criar 4 construtores:
  + Sem nenhuma informação passada (o time padrão será 0:0:0)
  + Passando somente a hora inicial (o time padrão será hora:0:0)
  + Passando somente a hora e minutos iniciais (o time padrão será hora:minutos:0)
  + Passando a hora, minutos e segundos iniciais (o time padrão será hora:minutos:segundos)
* Criar um método, toString(), que retorne uma string contendo uma hora no formato h:m:s

2. Criar a classe Circulo que possua:

* + Atributos inteiros x e y, que indicam as coordenadas do círculo na tela
  + Atributo real raio, que representa o tamanho do raio do círculo
  + Método para calcular a área (Área = Pi \* raio2) – utilize a constante Math.*PI* e a função Math.*pow(base, expoente)*, da classe Math
  + Método para calcular o perímetro do círculo (Perímetro = 2 \* Pi \* raio)
  + Métodos (moveX e moveY) que recebam como parâmetro um inteiro, e que mova o círculo horizontalmente e verticalmente
    - void moveX(int x1) – irá acrescentar x1 unidades à coordenada x
    - void moveY(int y1) – irá acrescentar y1 unidades à coordenada y
  + Método aumenta, que recebe como parâmetro um inteiro e aumenta o raio em função deste inteiro
    - void aumenta(double r) – o raio será multiplicado r vezes
  + Método exibe, que retorna as seguintes informações do círculo
    - Centro: (*x*, *y*)
    - Raio: *raio*
    - Área: *area*
    - Perímetro: *perimetro*
  + Crie um construtor que receba como parâmetro as coordenadas e o raio do círculo.
  + Crie um construtor vazio, que inicialize os valores das coordenadas e do raio com o valor 0
  + Para testar a sua classe, solicite inicialmente ao usuário que informe as coordenadas e o raio do círculo
  + Em seguida, exiba um menu com as seguintes opções:
    - 1 – Mover
    - 2 – Aumentar
    - 3 – Imprimir
    - 4 – Sair

3. Crie uma classe **Produto** com as seguintes características:

* Atributos id, nome e preço
* Um atributo ultimo\_id, que irá armazenar o identificador do último produto criado. Ao criar um novo produto, atribua o valor deste atributo ao id do objeto criado, e incremente este contador em seguida. Este atributo deverá ser somente de leitura fora da classe.
* Métodos get e set para acesso aos atributos da classe (para o atributo id, crie somente o método get. O valor deste atributo será alterado somente ao se criar um objeto da classe. Para o método setPreco, valide o valor do preço. Caso seja negativo, atribua o valor 0 ao mesmo)
* Construtor sem parâmetros, que atribua o id ao produto criado e que incremente o último id gerado, e que inicialize o preço do produto com 0 e o seu nome como “Nome não informado”
* Construtor que receba o nome e preço do produto, e que execute o primeiro para atribuir o valor do id produto
* Método toString(), que retorne uma string contendo as informações do produto da seguinte forma:
  + Produto: nome do produto - Id: identificador do produto - Nome: nome do produto
* Método reajustaPreco, que receba um percentual, e que atualize o preço do produto de acordo com o percentual
* Para testar a sua classe, crie dois objetos desta classe. Para a criação desses objetos, solicite ao usuário que informe o nome e preço do produto. Em seguida, solicite o valor de reajuste e imprima as informações do produto, juntamente com a informação do último id gerado.

4. Crie uma classe **Inteiro** para manipulação de valores inteiros. Esta classe deverá possuir os seguintes métodos estáticos:

* int multiplicacao(int a, int b): retorna a multiplicação de *a* por *b* (utilize somente a operação de soma para este cálculo)
* int potencia(int a, int b): retorna *a* elevado a *b* (utilize o método multiplicacao para este cálculo)
* int divisao(int a, int b): retorna o valor inteiro da divisão de *a* por *b* (utilize somente a operação de subtração para este cálculo)
* boolean par(int a): retorna *true* se *a* for par; e *false*, caso contrário (utilize o método resto para este cálculo)

Para testar a sua classe, exiba o seguinte menu para o usuário:

1 – Multiplicação

2 – Potência

3 – Divisão

4 – Par

5 – Sair

De acordo com a opção, solicite ao usuário os valores para o cálculo

5. Crie uma classe chamada Pessoa com os seguintes atributos: nome, idade, altura e peso.

* Considere, como subclasse da classe Pessoa, a classe Empregado. Considere que cada instância da classe Empregado tem, para além dos atributos que caracterizam a classe Pessoa, os atributos numeroSecao, salarioBase(vencimento base) e INSS (percentagem a ser retida do salário para pagar o INSS). Implemente a classe Empregado com um método calcularSalario.
* Considere, como subclasse da classe Pessoa a classe Cliente. Considere que cada instância da classe Cliente tem, para além dos atributos que caracterizam a classe Pessoa, os atributos credMax (correspondente ao crédito máximo concedido ao cliente) e valorEmDivida. Implemente na classe Cliente um método obterSaldo que devolve a diferença entre os valores dos atributos credMax e valorEmDivida.
* Implemente a classe Administrador como subclasse da classe Empregado. Um determinado administrador tem como atributos, para além dos atributos da classe Pessoa e da classe Empregado, o atributo ajudaDeCusto (ajudas referentes a viagens, estadias, ...). Note que deverá redefinir na classe Administrador o método herdado calcularSalario (o salário de um administrador é equivalente ao salário de um empregado usual acrescido das ajudas de custo).
* Implemente a classe Operario como subclasse da classe Empregado. Um determinado operário tem como atributos, para além dos atributos da classe Pessoa e da classe Empregado, o atributo valorProducao (que corresponde ao valor monetário dos artigos efetivamente produzidos pelo operário) e comissão (que corresponde à percentagem do valorProducao que será adicionado ao salário base do operário). Note que deverá redefinir nesta subclasse o método herdado calcularSalario (o salário de um operário é equivalente ao salário de um empregado usual acrescido da referida comissão).
* Implemente a classe Vendedor como subclasse da classe Empregado. Um determinado vendedor tem como atributos, para além dos atributos da classe Pessoa e da classe Empregado, o atributo valorVendas (correspondente ao valor monetário dos artigos vendidos) e o atributo comissão (percentagem do valorVendas que será adicionado ao salarioBase do Vendedor). Note que deverá redefinir nesta subclasse o método herdado calcularSalario (o salário de um vendedor é equivalente ao salário de um empregado usual acrescido da referida comissão).
* Para cada uma das classes implementadas, crie também um método get e set para seus atributos, bem como um método toString() que retorne as informações da classe.
* Escreva um programa de teste adequado para as classes implementadas.