Lista de Exercícios POO

1. Faça uma Classe chamada Mae, que receba atributos privados de nome, idade, data de nascimento e altura.

Crie um `Constructor()` para a classe Mae, passando os atributos da classe Mae para o `Constructor()`

Crie o Método `GetNameMae()` que retorne o nome da Mãe

Crie o Método `SetNameMae()` que recebe um nome pelo parâmetro, e altera o atributo Nome da Classe Mae

Crie o Método `GetIdadeMae()` que retorne a idade da Mãe

Crie o Método de `FazerAniversario()` onde a mãe irá fazer um aniversário e incrementar a idade dela.

Crie o Método `Apresentar()` onde a mãe se apresenta, utilizando o nome dela, a data de nascimento dela, e a altura dela

Na **Main()**, Instancie um objeto vindo da classe Mae, passando o Nome da mãe, a idade da mãe, a data de nascimento da mãe, e altura da mãe.

Requisitos:

(1) Mostre no Console, uma apresentação da Mãe, informando os atributos que ela possui na classe dela.

(2) Faça a idade da mãe incrementar 3 vezes.

(3) Adicione um sobrenome ao nome da mãe e mostre no console também.

2. Crie uma classe **Calculadora,** esta classe deve ter os seguintes métodos:

a. **Soma**

b. **Subtração**

c. **Divisão**

d. **Multiplicação**

Requisitos:

- Utilizando herança crie uma classe chamada Calculadora Cientifica que contém os métodos para calcular **raizQuadrada** e a **potencia.**

- No **Main()** instancie um objeto Calculadora e CalculadoraCientifica

- No console, exiba o resultado de 4 operações matemáticas contendo **Soma, Subtração, Divisão, Multiplicação**

- Ainda no console, exiba o resultado de 2 cálculos utilizando **raizQuadrada e potencia**

3. Crie uma classe **AgendaIma** que irá conter alguns métodos dentro dela para poder armazenar algumas pessoas na agenda da imã e salvar alguns dados na agenda:

a. Crie um método chamado ArmazenarPessoaNaAgenda() para armazenar uma pessoa dentro de um array, e esse método deve conter no parâmetro informações sobre a pessoa armazenada.

b. Crie um método chamado RemoverPessoaDaAgenda() remover uma pessoa dentro do array criado, você deve remover a pessoa baseado apenas no nome dela.

c. Crie um método chamado BuscarPessoaNaAgenda() onde você deve informar qual posição no array essa pessoa está, baseado no array onde ela está armazenada.

d. Crie um método chamado ImprimirAgenda() , onde deve imprimir os dados das pessoas que estão no array

e. Crie um método chamado ImprimirPessoa() , onde vai receber o nome da pessoa no parâmetro e imprimir os dados da pessoa.

Requisitos:

- No Main(), instancie uma agenda da imã, e faça o teste dos métodos criados.

- Utilize todas essas operações dos métodos criados, exibindo no console o resultado delas.

4. Para representar funcionários em uma empresa, crie uma classe chamada Funcionario que inclua três **propriedades**:

a. **Propriedade** nome que possui um Get e um set

b. **Propriedade** sobrenome que possui um Get e um set

c. **Propriedade** salário mensal que possui um Get e um set

- A classe funcionario precisar ter um construtor que inicializa os três atributos.

Forneça os métodos getters e setters para cada atributo.

- Dentro da **propriedade** nome e sobrenome, altere os getters dessas propriedades, para retornarem o nome do usuário.

- Dentro da **propriedade** salario mensal, altere o setter dela verificando se o salário é menor que zero, se for, exiba um erro e adicione o valor 0 a **propriedade** salario mensal, se não, adicione um novo valor no setter dessa propriedade.

Requisitos:

- Crie duas instâncias(**objetos**) da classe Funcionario

- Exiba o nome do funcionario junto com o sobrenome e o salario mensal deles.

- Dê a cada funcionario um aumento de 20% e exiba dessa vez o salário anual de cada funcionario.

5. Crie uma classe abstrata chamada **FiguraGeometrica** que vai servir de herança para outras classes.

Essa classe abstrata **FiguraGeometrica** deve conter uma única assinatura chamada **CalcularArea()**

- Crie uma pasta Models e dentro dela crie as seguintes classes:

a. Classe Triângulo - contendo as propriedades base e altura

b. Classe Quadrado - contendo a propriedade lado do quadrado

c. Classe Trapézio - contendo as propriedades base maior, base menor e altura

- Cada uma dessas classes devem possuir a herança da classe abstrata, e cada uma dessas classes devem reeimplementar o método de **CalcularArea()**

Requisitos:

- Instancie um triangulo, um quadrado e um trapezio das classes criadas.

- Mostre no Console o valor da **Área do Triangulo**

- Mostre no Console o valor da **Área do Quadrado**

- Mostre no Console o valor da **Área do Trapezio**

6. Crie uma classe Abajur que possui um atributo ligado, que indica se o Abajur está ligado ou desligado.

- Crie um método para Ligar o Abajur

- Crie um método para Desligar o Abajur

- Crie um método imprimir, para imprimir no console se o abajur está ligado ou não.

Requisitos:

- Ao construir(**instanciar**) um abajur, o estado dele (**ligada ou desligada**) deve ser fornecido.

- Para **ligar** e **desligar** o abajur, os métodos `Ligar()` e `Desligar()` devem ser chamados.

Essa é a única forma de alterar o estado do abajur, já que o atributo dele não pode ser acessado de fora da classe.

- Imprima no console:

"Abajur ligada" ou "Abajur desligado", dependendo do estado atual.

7. (Desafio) A imã precisa de uma nota fiscal para cada cliente que compre algum produto dela.

- Crie uma Classe Cliente, essa classe deve ter um construtor, algumas propriedades.

    Construtor() Contendo as propriedades NomeCliente, CpfCliente, SaldoDoCliente

    Propriedades

    1. NomeCliente (String)
    2. CpfCliente (String)
    3. SaldoCliente (Decimal)

- Crie uma Classe Produto, essa classe deve ter um construtor, algumas propriedades e métodos.

    Construtor() Contendo as propriedades NomeProduto , IdProduto, ValorProduto

    Propriedades

    1. NomeProduto (String)
    2. IdProduto (String)
    3. ValorProduto (Decimal)

    Metodos

    1. Crie um método chamado ComprarProduto(), que recebe no parâmetro um tipo de dado Cliente, e o NomeDoProduto, e ValorProduto e verifique dentro do método ComprarProduto(), se o cliente tem o saldo positivo, e se da pra comprar o produto escolhido.

    2. Crie um método chamado GerarNotaFiscal(), dentro desse método verifique se a validação o método ComprarProduto() deu certo, se deu certo você deve chamar uma instância de NotaFiscal.

- Com o objeto NotaFiscal criado, você deve passar pro objeto NotaFiscal o nome do Produto, o Id Do Produto e o valor do produto.

- Em seguida pode acessar um método privado de NotaFiscal que vai gerar uma sequência de hashCode podendo ser em string.

- Crie uma Classe NotaFiscal, essa classe pode fazer herança com a classe Produto.

Construtor() Contendo as propriedades NomeProduto , IdProduto, ValorProduto

Metodos

1. Crie um método chamado MostrarNotaFiscal(), que pode ser privado, e retorna o Nome e o valor do produto, junto com um hash code em string