PYTHON ORIENTADO A OBJETOS

Exercício 1: Carro Elétrico

Crie uma classe Carro com os seguintes atributos:

- marca (a marca do carro)
- modelo (o modelo do carro)
- ano (o ano de fabricação do carro)
- quilometragem (a quilometragem do carro)

A classe Carro deve ter os seguintes métodos:

- informacoes(): que imprime todas as informações do carro.
- andar(distancia): que simula o carro andando, aumentando a quilometragem de acordo com a distância percorrida.

Em seguida, crie uma classe CarroEletrico que herde da classe Carro e tenha um atributo adicional:

- autonomia (a autonomia do carro elétrico em quilômetros)

A classe CarroEletrico deve ter um método adicional:

- informacoes(): que sobrescreve o método informacoes() da classe Carro, incluindo informações sobre a autonomia do carro elétrico.

Exercício 2: Produto Importado

Crie uma classe Produto com os seguintes atributos:

- nome (o nome do produto)
- preco (o preço do produto)
- estoque (a quantidade disponível em estoque do produto)

A classe Produto deve ter o seguinte método:

- informacoes(): que imprime todas as informações do produto.
- vender(quantidade): que simula a venda do produto, reduzindo a quantidade em estoque.

Em seguida, crie uma classe ProdutoImportado que herde da classe Produto e tenha um atributo adicional:

- pais_origem (o país de origem do produto importado)

A classe ProdutoImportado deve ter um método adicional:

- informacoes(): que sobrescreve o método informacoes() da classe Produto, incluindo informações sobre o país de origem.

Exercício 3: Conta Corrente

Crie uma classe ContaBancaria com os seguintes atributos:

- titular (o nome do titular da conta)
- saldo (o saldo da conta)

A classe ContaBancaria deve ter os seguintes métodos:

- depositar(valor): que adiciona um valor ao saldo da conta.
- sacar(valor): que subtrai um valor do saldo da conta, desde que haja saldo suficiente.
- obter_saldo(): que retorna o saldo atual da conta.

Em seguida, crie uma classe ContaCorrente que herda da classe ContaBancaria e tenha um atributo adicional:

- limite_cheque_especial (o limite do cheque especial da conta corrente)

A classe ContaCorrente deve ter um método adicional:

- usar_cheque_especial(valor): que permite ao titular usar o cheque especial caso o saldo seja insuficiente para um saque.