Nível Iniciante: Criando os Primeiros Moldes

O foco aqui é se acostumar com a sintaxe básica: criar uma classe, definir o __init__, criar atributos e instanciar objetos.

Exercício 1: Molde de uma Pessoa

Crie uma classe chamada Pessoa. No "registro de nascimento" (__init__), toda pessoa deve ter um nome e uma idade.

Exercício 2: Criando Pessoas Reais

Usando a classe Pessoa que você criou, crie dois objetos:

1. Uma pessoa chamada "João", com 25 anos.
2. Uma pessoa chamada "Maria", com 30 anos.
   Depois de criá-los, imprima o nome e a idade de cada um para confirmar que deu certo.

Exercício 3: Ensinando a se Apresentar

Adicione um método (uma "ação") à sua classe Pessoa chamado apresentar. Esse método deve imprimir uma frase como: "Olá, meu nome é [nome] e eu tenho [idade] anos." Chame esse método para os objetos "João" e "Maria".

Exercício 4: Molde de um Produto

Crie uma nova classe chamada Produto. Todo produto deve ter nome e preço. Depois, crie duas instâncias:

1. Um "Caderno" que custa 15.50.
2. Uma "Caneta" que custa 3.00.
   Imprima o nome e o preço de cada produto.

Nível Intermediário: Objetos que Mudam e Calculam

Aqui, os objetos começarão a ter métodos que alteram suas próprias características (atributos) e que realizam cálculos.

Exercício 5: Conta Bancária

Crie uma classe ContaBancaria. Toda conta deve começar com um titular e um saldo inicial. Crie um método depositar(valor) que some um valor ao saldo da conta. Crie um objeto, deposite um valor e imprima o novo saldo.

Exercício 6: Lógica no Saque

Na mesma classe ContaBancaria, adicione um método para sacar(valor). Este método deve verificar se há saldo suficiente na conta.

- Se houver, ele deve subtrair o valor do saldo e imprimir "Saque realizado com sucesso.".
- Se não houver, ele deve imprimir "Saldo insuficiente." e não alterar o saldo.
  Teste os dois cenários: um saque bem-sucedido e uma tentativa de sacar mais do que tem.

Exercício 7: Calculadora de Retângulos

Crie uma classe Retangulo que é inicializada com base e altura. Crie dois métodos:

1. calcular_area(): deve retornar o cálculo base * altura.
2. calcular_perimetro(): deve retornar o cálculo 2 * (base + altura).
   Crie um retângulo, chame os dois métodos e imprima os resultados que eles retornam.

Exercício 8: Um Carro que Anda

Crie uma classe Carro com os atributos modelo e nivel_combustivel (começando com 0).

1. Crie um método para abastecer(litros) que aumenta o nível de combustível.
2. Crie um método dirigir(distância) que consome combustível (ex: 1 litro a cada 10 km). O método deve verificar se há combustível suficiente para a viagem. Se houver, diminua o combustível e avise que o carro andou. Se não, avise que não há combustível.
3.

Nível Avançado: Interação e Organização

Estes exercícios envolvem como as classes e objetos podem interagir entre si e conceitos um pouco mais complexos.

Exercício 9: Atributo da Classe (Funcionários da Empresa)

Crie uma classe Funcionario. Cada funcionário terá nome e salário (atributos de instância). Agora, crie um atributo de classe chamado percentual_bonus, com o valor 1.10 (representando 10% de bônus).
Crie um método aplicar_bonus que multiplica o salário do funcionário pelo percentual_bonus da classe. Crie dois funcionários com salários diferentes, aplique o bônus e veja o resultado.

- **Dica:** Um atributo de classe é definido diretamente dentro da classe, fora de qualquer método.

Exercício 10: Um Objeto Dentro de Outro

Crie duas classes: Motor e Carro.

1. A classe Motor terá um atributo potencial.
2. A classe Carro terá modelo. Ao criar um Carro, ele deve também criar um objeto Motor e guardá-lo como um de seus atributos (ex: self.motor = Motor(100)).
   Crie um método no Carro chamado exibir_potencia que imprime a potência do seu motor.

Exercício 11: Biblioteca de Livros

Crie duas classes: Livro e Biblioteca.

1. A classe Livro terá atributos título e autor.
2. A classe Biblioteca terá um atributo acervo, que começa como uma lista vazia [].
3. A Biblioteca deve ter dois métodos:
    - adicionar_livro(livro): recebe um **objeto** Livro e o adiciona à lista acervo.
    - listar_livros(): percorre a lista acervo e imprime o título e o autor de cada livro.

Crie uma biblioteca, crie alguns objetos Livro e adicione-os à biblioteca. Depois, liste os livros.

Exercício 12: Representação Amigável (__str__)

Crie uma classe Filme com título, diretor e ano. Se você tentar dar print() em um objeto Filme, o resultado não será muito útil. Para resolver isso, implemente o método especial __str__(self). Este método deve retornar uma string formatada, como por exemplo: "Filme: 'De Volta para o Futuro' (1985) - Diretor: Robert Zemeckis".
Depois de implementar, crie um objeto Filme e simplesmente use print(meu_filme).

- **Dica:** O que o método __str__ retorna (return) é o que será exibido quando o objeto for impresso.