



Programação Orientada a Objetos em Python

Semana 3 / Atividade 3

Nesta atividade, você vai dar os primeiros passos com Programação Orientada a Objetos (POO) em Python. A ideia é sair de programas estruturados em funções e começar a modelar o problema usando classes, atributos, métodos, propriedades e relacionamentos entre objetos.

Observação importante sobre os vídeos

Nos vídeos, a classe **Retangulo** é escrita como **Rentangulo** por engano (*typo*).

Esse erro é mantido até o vídeo 5, quando é identificado e corrigido.

No seu código, você pode usar o nome correto **Retangulo** desde o vídeo 1 😊.

Objetivos

Ao final deste estudo dirigido, você deverá:

- Entender os conceitos básicos de Programação Orientada a Objetos em Python.

- Criar classes com atributos de instância e atributos de classe.
- Definir métodos de instância e usar o parâmetro self.
- Utilizar propriedades para validar e proteger atributos.
- Implementar métodos especiais (como `__str__`, `__repr__`, `__eq__`) para melhorar o comportamento das classes.
- Modelar relacionamentos entre classes, como um projeto contendo várias participações de alunos.

Atividades

Acompanhar as aulas e codar junto

Assista às aulas abaixo fazendo os exemplos e testes junto com o instrutor no VS Code, salve no diretório `src/07-orientacao-a-objetos`

- Aula 07.01 – Introdução à Orientação a Objetos
 - <https://www.youtube.com/watch?v=iLWsDklMHTc>
- Aula 07.02 – Atributos de instância e de classe
 - <https://www.youtube.com/watch?v=CF5Zdm7QHMK>
- Aula 07.03 – Métodos de classe
 - <https://www.youtube.com/watch?v=awQkzLXeA5U>
- Aula 07.04 – Propriedades
 - <https://www.youtube.com/watch?v=SGDx-FjEj8Q>
- Aula 07.05 – Métodos especiais
 - <https://www.youtube.com/watch?v=Fnb2sq7fCsg>
- Aula 07.06 – `__eq__` e `__hash__`
 - <https://www.youtube.com/watch?v=yMlgbYBGMWA>

- Aula 07.07 – Relacionamento entre classes
 - https://www.youtube.com/watch?v=vG_P6Y5qes0
- Herança entre Classes – super()
 - <https://www.youtube.com/watch?v=EEmVxcu0mAU>

Exercícios Práticos

Dentro do diretório `src/07-orientacao-a-objetos/exercicios`, implemente os programas abaixo.

- `ex01.py` – Classe Aluno
 - Implemente uma classe Aluno com:
 - Atributos:
 - `prontuario`
 - `nome`
 - `email`
 - Requisitos:
 - Deve ser possível construir um objeto Aluno a partir de uma string no formato: `"prontuario,nome,email"`
Exemplo: `"SP0101,João da Silva, joao@email.com"`
 - Nenhum dos atributos pode ser vazio ou nulo.
 - Use propriedades (`@property` e `setters`) para validar os valores.
 - Dois alunos devem ser considerados iguais se tiverem o mesmo prontuário.
 - Implemente o método especial `__eq__` para comparar objetos Aluno por prontuário.

- (Opcional, mas recomendado) Considere também implementar `__hash__` se quiser usar alunos em conjuntos (set) ou como chaves de dicionário.
- ex02.py – Classe Projeto
 - Implemente uma classe Projeto com:
 - Atributos:
 - codigo – número inteiro que representa o código do projeto
 - titulo
 - responsavel – nome do professor responsável pelo projeto
 - Requisitos:
 - Deve ser possível construir um objeto projeto a partir de uma string no formato: "codigo,titulo,responsavel"
Exemplo: "1,Laboratório de Desenvolvimento de Software,Pedro Gomes"
 - Nenhum dos atributos pode ser vazio ou nulo (use propriedades).
 - O atributo codigo deve ser armazenado como inteiro.
 - Dois projetos devem ser considerados iguais se tiverem o mesmo código (codigo).
 - Implemente `__eq__` comparando pelo código.
- ex03.py – Classe Participacao
 - Implemente uma classe Participacao com os seguintes atributos:

- código – identificador da participação (pode ser inteiro ou string, você escolhe, mas seja consistente)
- data_inicio
- data_fim
- aluno – um objeto da classe Aluno
- projeto – um objeto da classe Projeto associado
- Você pode começar armazenando as datas como strings (ex: "2025-03-01"). Em atividades futuras, podemos trabalhar com tipos de data mais específicos.
- ex04.py – Lista de participações no Projeto
 - Altere a classe Projeto para:
 - Incluir um atributo do tipo list chamado participacoes.
 - Essa lista deve armazenar objetos da classe Participacao.
 - Implemente também o método:

```
def add_participacao(self, participacao):  
    """Adiciona uma participação ao projeto."""  
    ...
```
 - Requisitos:
 - O método add_participacao deve receber um objeto Participacao como parâmetro e adicioná-lo à lista participacoes.

Entrega da Atividade

Gerar um arquivo PDF que contenha:

- Seu nome.
- O link do seu repositório no Github.
- O código de todas as atividades desse roteiro de estudos (código das videoaulas, reflexões e exercícios)

Enviar o arquivo PDF na tarefa S3.A3. Programação Orientada a Objetos em Python.