Aula 01 – Introdução a POO

TECNOLOGIA em **Gestão de Dados**

Programação II

Prof. Dr. Wener Sampaio

ReUni DIGITAL · EAD Centro de Educação Aberta e a Distância · UFPI Universidade Federal do Piauí · MEC

Programação Orientada a Objetos

- Em Python:
  - É uma linguagem de programação multiparadigma: imperativa, funcional e também orientada a objetos.
  - Mantém a simplicidade também em OO.

Programação Orientada a Objetos

- Foco nos dados (classes/objetos), ao invés das funções.
- As funções mudam mais do que os dados.
- Permitem recursos como:
  - Reutilização
  - Modularização
  - Herança
  - Polimorfismo
  - Interfaces
  - Classes abstratas
  - Encapsulamento

Programação Orientada a Objetos

- POO é baseada no conceito de classes e objetos.
- Classes agrupam dados e código:
  - Dados (variáveis) -> atributos
  - Código (funções) -> métodos
  - **Definem** as características e o comportamento dos objetos, mas **não armazenam dados nem executam comandos**.
  - São semelhantes aos tipos: int, str, bool, list, etc.
    - dir(str)
    - type('texto')

```
class Cachorro:
    def __init__(self, raca, cor):
        self.raca=raca
        self.cor=cor

    def latir(self, qnt=1):
        print('Au! '*qnt)
```

Programação Orientada a Objetos

- Objetos são instâncias das classes:
  - Um objeto é a “materialização” de uma classe.
  - Executam os métodos e armazenam dados sobre/nos atributos.
  - São semelhantes às variáveis:
    - x=5 # criação por atribuição
    - teste= x > 10 # uso da variável x
    - y = float(“3.1415”) # uso de uma função externa
    - bilu = Cachorro(‘Pastor alemão ’, ‘marrom’) #criação por atribuição
    - print(bilu.raca, bilu.cor) # uso dos atributos
    - bilu.latir(3) #uso do método, “função interna”

```
class Cachorro:
    def __init__(self, raca, cor):
        self.raca=raca
        self.cor=cor

    def latir(self, qnt=1):
        print('Au! '*qnt)
```

Programação Orientada a Objetos

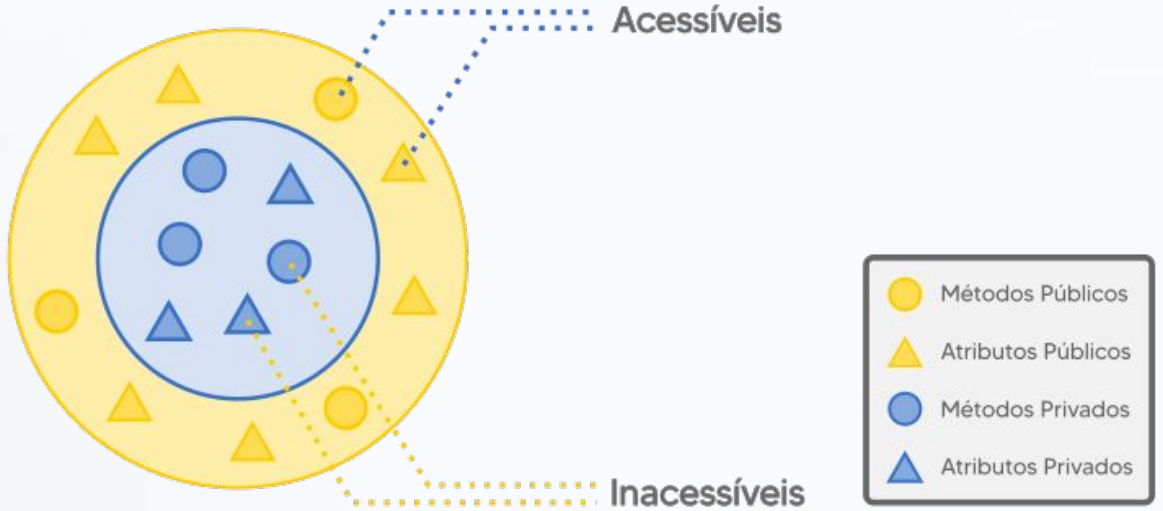
- Redefinindo:
  - As Classes são tipos de dados definidos pelo desenvolvedor que atuam como um modelo para objetos.
    - Classes -> formas de bolo
    - Objetos -> bolos
  - Não existe um objeto sem uma Classe que o defina.
  - Um Classe sem objeto é inútil.

Programação Orientada a Objetos

- Conceitos básicos – Encapsulamento:
  - Serve para juntar atributos e métodos em uma única entidade.
  - Ao se criar um objeto todos os atributos e métodos já estão disponíveis, isolados e protegidos (ou não).
    - lulu = Cachorro('pitbull', 'creme')
    - kratos = Cachorro('pinscher', 'branco')

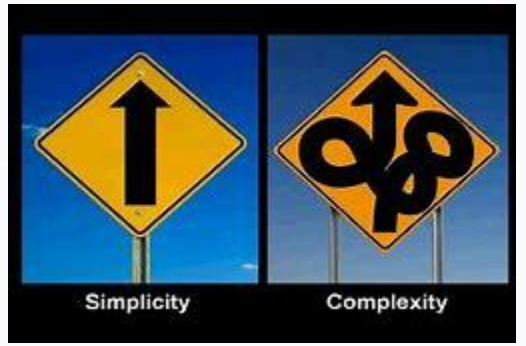
Programação Orientada a Objetos

- Conceitos básicos – Encapsulamento:
  - Alguns métodos e atributos são perigosos, sigilosos ou importantes.
    - Não é interessante dar acesso a qualquer outro objeto, classe ou trecho de código.
    - Apenas a própria classe/objeto (e alguns outros) pode ter acesso a tais recursos.



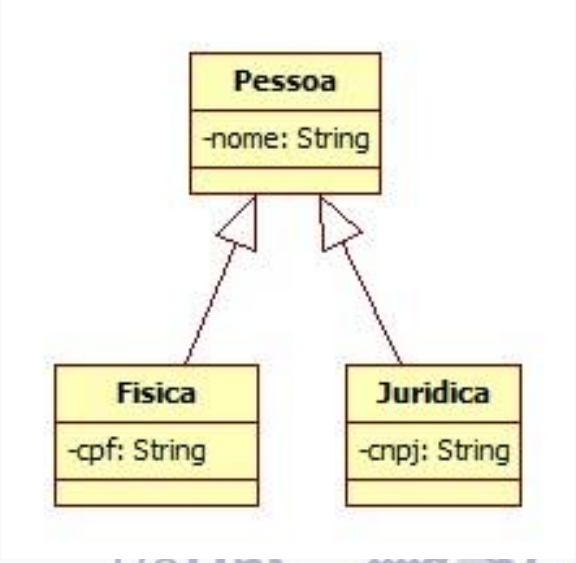
Programação Orientada a Objetos

- Conceitos básicos – abstração:
  - Em programação o termo ‘abstrair’ significa esconder, simplificar.
  - Só mostrar o que realmente for necessário
  - A complexidade fica escondida.
  - Sabemos o que faz, mas não sabemos como é feita.
  - Podemos trocar toda a lógica e manter a funcionalidade
  - Ex.:
    - impressoraG3000.imprimir()
    - impressoraG3100.imprimir()
    - impressoraG3110.imprimir()



Programação Orientada a Objetos

- Conceitos básicos – herança
  - Permite a reutilização de código
  - Uma Classe pode herdar métodos e atributos da classe pai.
    - Classe herda da superclasse.
  - Ex:
    - p1 = Fisica()
    - p1.nome='Astrobaldo'
    - p1.cpf='000.111.222-33'
    - p1.cnpj='00.111.222/0001-33' #erro!



Programação Orientada a Objetos

- Conceitos básicos – polimorfismo
  - Conceito relacionado à herança.
  - Uma classe filha pode herdar um método do pai, mas realizar uma tarefa diferente.