Introdução às Classes

Programação Orientada a Objetos

Python

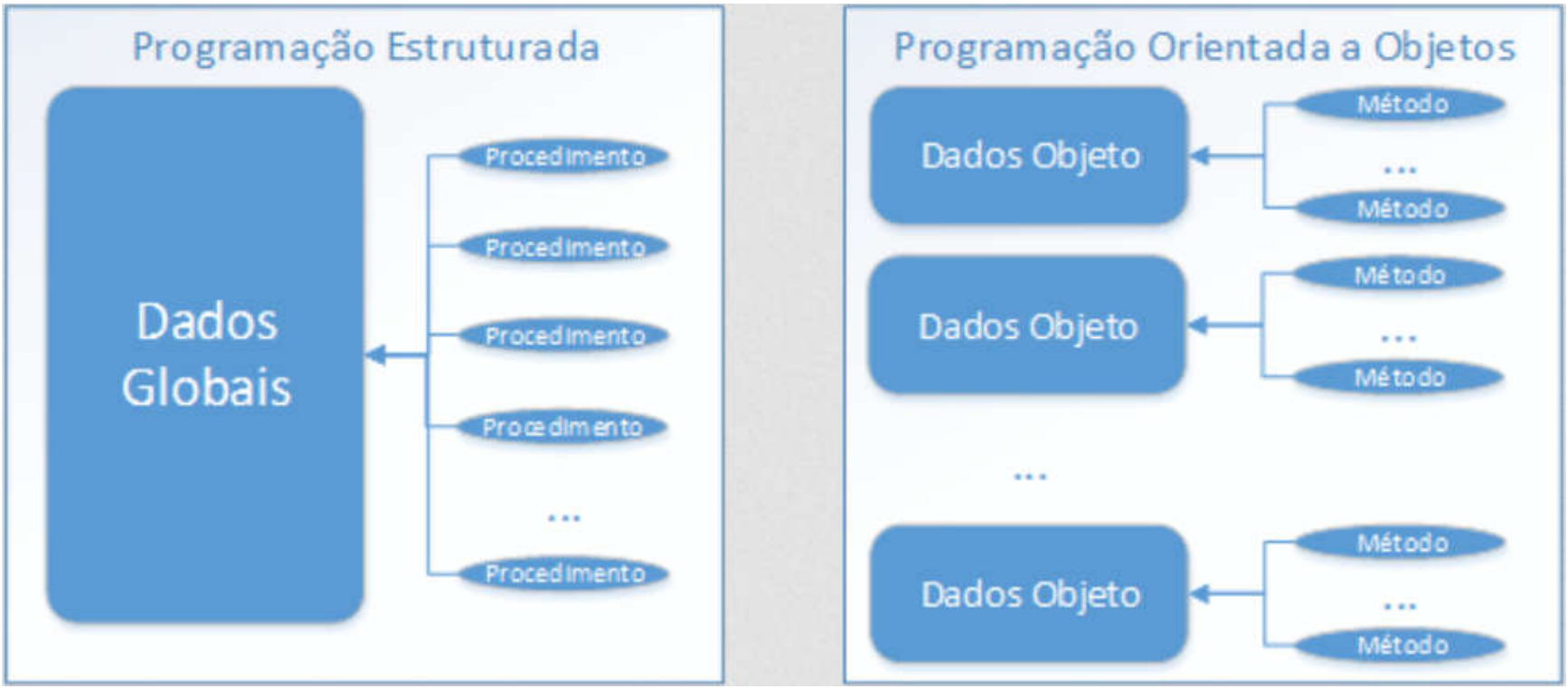
Prof. Diógenes Furlan

PE x POO

- Programação Estruturada
  - Enfoque nos processos
  - Dados globais / locais
  - Abstração

- Programação Orientada a Objetos
  - Enfoque nos dados
  - Dados globais / locais / classe
  - Encapsulamento
  - Herança
  - Polimorfismo

PE x POO



Fonte: http://www.devmedia.com.br/os-4-pilares-da-programacao-orientada-a-objetos/9264

Classe

- Uma classe é um tipo definido pelo usuário que contém o molde, a especificação para os objetos (modelo).
  - Definir uma classe envolve trabalhar com atributos (dados) e funções (código).
- Conceito idêntico ao de **Estrutura / Registro**.
- Tem a função de definir um novo **Tipo**.

Conceitos Básicos da Aula

- Classe
- Objeto
- Atributo
- Propriedade
- Método
- Mensagem

Objeto

- Um objeto é uma instancia de uma classe.
- Conceito idêntico ao de **Variável.**

- Objetos podem modelar entidades do mundo real (Carro, Pessoa, Usuário) ou entidades abstratas (Temperatura, Umidade, Medição, Configuração).

Atributo

- Atributos são as **variáveis** “locais” da classe

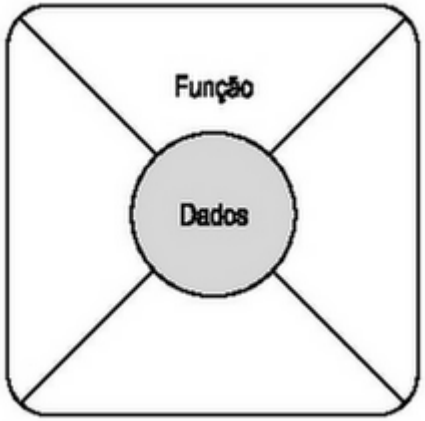
- Um atributo é chamado de propriedade quando:
  - é um atributo “privado”
  - possui métodos controladores “get” e “set”

Método

- Métodos (ou funções membro) são as "**rotinas**" que utilizam aquela estrutura
- Quando uma função membro é chamada, se diz que o objeto está recebendo uma mensagem (para executar uma ação).

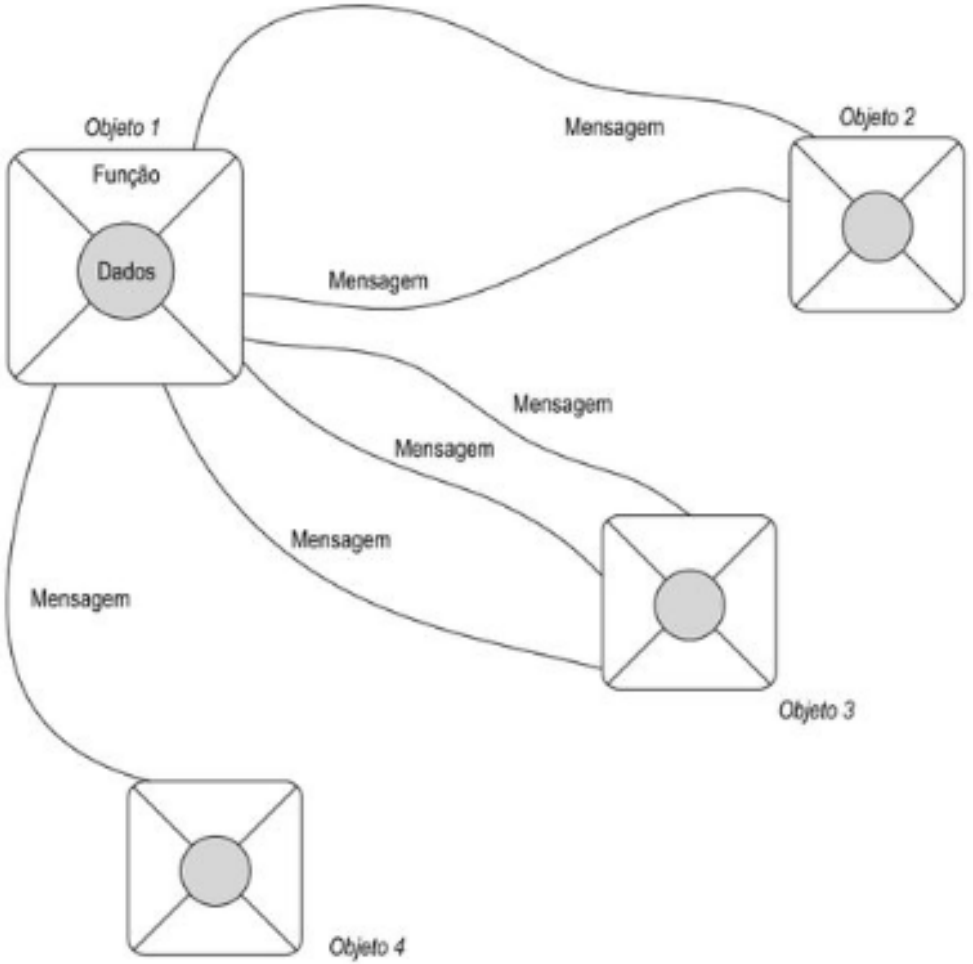
- Podem ser descritos inline ou outline

Objeto encapsulado



Fonte: Lee, UML e C++

Passagem de mensagens entre objetos



Fonte: Lee, UML e C++

Diagrama UML de Classe

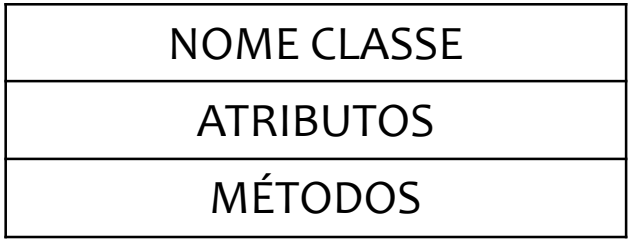
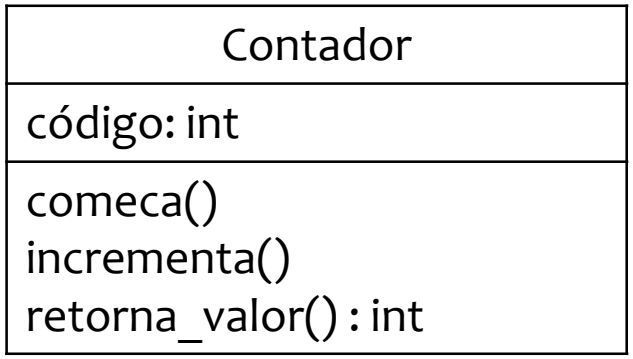
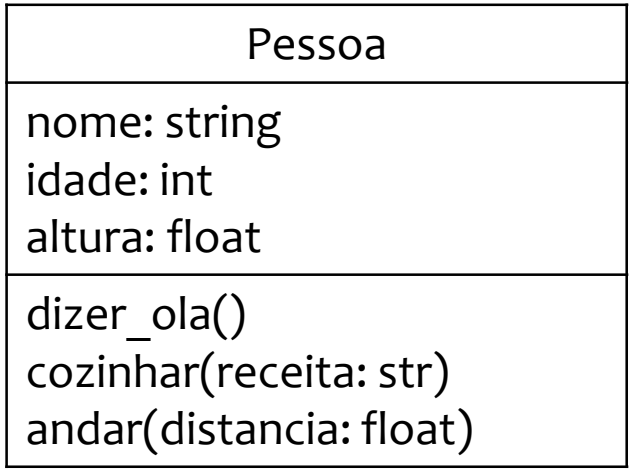
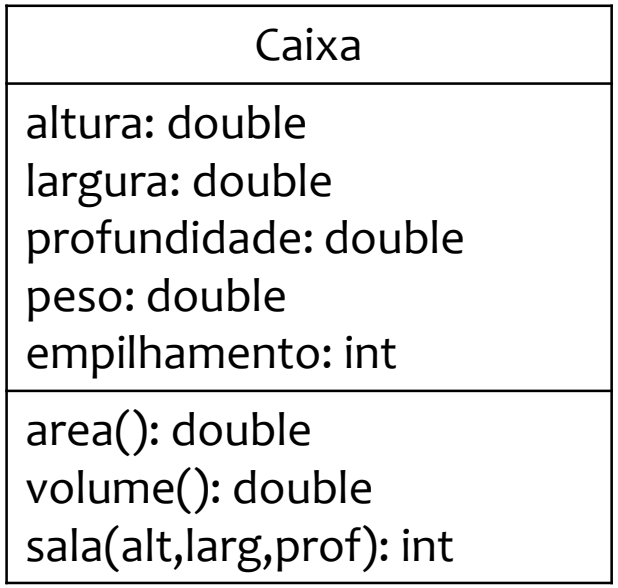


Diagrama UML de Classe





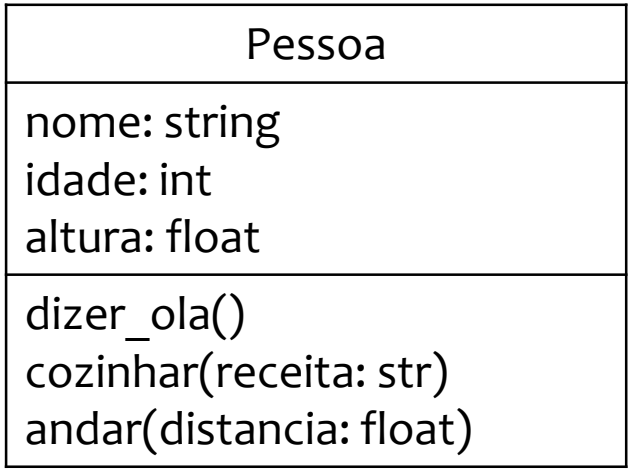


POO no Python

- Python possui palavras reservadas (*keywords*) para criarmos **Classes** e **Objetos**.
  - *keyword* class – utilizada para criar uma classe.
  - *keyword* self – utilizada para guardar a referência ao próprio objeto.

Exemplo

- Vamos criar uma classe que representa uma entidade do tipo **Pessoa**!
- Ela deve ter os seguintes campos:
  - Nome como String;
  - Idade como Inteiro;
  - Altura como Decimal.
- E deve ter métodos para:
  - Dizer “Olá”;
  - Cozinhar;
  - Andar.



Exemplo – Classe

```
class Pessoa:
    def __init__(self, nome: str, idade: int, altura: float):
        self.nome = nome
        self.idade = idade
        self.altura = altura

    def dizer_ola(self):
        print(f'Olá, meu nome é {self.nome}. Tenho {self.idade} '
              f'anos e minha altura é {self.altura}m.')

    def cozinhar(self, receita: str):
        print(f'Estou cozinhando um(a): {receita}')

    def andar(self, distancia: float):
        print(f'Saí para andar. Volto quando completar {distancia} metros')
```

Exemplo – Programa

```
# Instancia um objeto da Classe "Pessoa"
pessoa = Pessoa(nome='Zhongli', idade=6000, altura=1.90)

# Chama os métodos de "Pessoa"
pessoa.dizer_ola()
pessoa.cozinhar('Spaghetti')
pessoa.andar(500)
```

Observações

- **self:** vai aparecer sempre como 1º parâmetro de métodos de classe
  - é uma referencia ao próprio objeto

- **__init__:** método chamado de **Construtor**.
  - é chamado ao se instanciar objetos.
  - é nele que “setamos” os atributos do objeto.

Exemplo 2

Contador
código: int
comeca() incrementa() retorna_valor() : int

```
class Contador:
    def __init__(self):
        self.codigo = 0

    def incrementa(self):
        self.codigo = self.codigo + 1

    def valor(self):
        return self.codigo
```

Programa 2

```
cont1 = Contador()
print( cont1.valor() )
cont1.incrementa()
print( cont1.valor() )
cont1.incrementa()
print( cont1.valor() )
```

EXERCÍCIOS

Exercício 1/3

Defina até 5 atributos e 5 métodos para cada uma das seguintes classes (mostre também seu Diagrama UML):

- Carro
- Computador
- Conta
- Maçã
- Cliente
- Funcionário
- Data
- Casa
- Retângulo
- TV

Exercício 2/3

- Programe a classe Caixa:

Caixa
altura: double largura: double profundidade: double peso: double empilhamento: int
area(): double volume(): double numCaixasSala(alt_sala, larg_sala, prof_sala): int

Exercício 3/3

- Programe uma das classes que você criou.