Composição

Programação Orientada a Objetos

Python

Prof. Diógenes Furlan

EXEMPLO 1

Exemplo 1

- Objetos podem ser compostos por outros objetos.

```
class Data:
    def __init__(self, d, m, a):
        self.dia = d
        self.mes = m
        self.ano = a

class Pessoa:
    def __init__(self, c, n, d:Data):
        self.codigo = c
        self.nome = n
        self.dtNasc = d

# main
d1 = Data(11,4,2024)
p1 = Pessoa(1,'Joana',d1)
```

Exemplo 1

- Inserindo métodos

```
class Data:
    def __init__(self, d=0, m=0, a=0):
        self.dia = d
        self.mes = m
        self.ano = a

    def input(self):
        self.dia = int(input('Entre com o dia: '))
        self.mes = int(input('Entre com o mês: '))
        self.ano = int(input('Entre com o ano: '))

    def __str__(self):
        return f'{self.dia:02}/{self.mes:02}/{self.ano:04}'
```

Exemplo 1

- Inserindo métodos

```
class Pessoa:
    def __init__(self, c=0, n='', d=Data()):
        self.codigo = c
        self.nome = n
        self.dtNasc = d

    def input(self):
        print('Pessoa')
        self.nome = input('Entre com o nome: ')
        self.codigo = int(input('Entre com o código: '))
        print('Data Nascimento')
        self.dtNasc.input()

    def __str__(self):
        aux = 'PESSOA\n======\n'
        aux += f'nome.............: {self.nome}\n'
        aux += f'data nascimento..: {self.dtNasc}'
        return aux
```

EXEMPLO 2

Exemplo 2

```
class Somador:
    def __init__(self, v=0):
        self.valor = v

    def soma(self,v):
        self.valor += v

    def getValor(self):
        return self.valor

    def __str__(self):
        print(f'Somador={self.valor}')
```

Exemplo 2

```
class Media:
    def __init__(self):
        self.soma = Somador()
        self.qtd = Somador()

    def adiciona(self,v):
        self.soma.soma(v)
        self.qtd.soma(1)

    def getMedia(self):
        return self.soma.getValor() / self.qtd.getValor()

    def __str__(self):
        aux = f'Total: {self.soma.getValor()}'
        aux += f'\nQuantidade: {self.qtd.getValor()}'
        aux += f'\nMédia: {self.getMedia()}'
        return aux
```

Exemplo Composição

```
# main
m1 = Media()
valor = 0

while valor >= 0:
    valor = float(input('Digite um valor positivo (negativo encerra): '))
    if valor > 0:
        m1.adiciona(valor)

print( m1 )
```

EXERCÍCIOS

Exercício 1

- Crie uma classe chamada Ponto.
  - com 2 atributos: X e Y
- Crie um construtor para esta classe:
  - com 2 parâmetros
  - e valores default = 0
- Crie um método chamado distancia:
  - que recebe outro Ponto

Exercício 2 (a)

- Crie uma classe Reta que tem como atributos dois objetos da classe Ponto.
- Como criar 2 construtores para esta classe?
  - um que inicializa os pontos a partir de 2 pontos passados por parâmetro
  - um que inicializa os pontos a partir de 4 coordenadas reais passadas por parâmetro

Exercício 2 (b)

- Crie uma função que calcula a distancia dos 2 pontos da reta. Reutilize código da classe Ponto.
- Mostre a reta, mostrando seus pontos e sua distancia.

Exercício 2 (c)

- Quando seu programa ficar pronto acrescente funções membros para esta reta, tais como:
  - coeficiente angular (inclinação) → a
  - coeficiente linear (corta eixo y) → b
- Faça uma rotina que mostra a reta na forma
  - y=ax + b
- Defina uma função membro para a classe reta que retorna o ponto de intercessão com outra reta
  - **ponto reta::intercessao(reta a);**.
  - Não se esqueça de tratar os casos degenerados, tais como retas paralelas e coincidentes, use por exemplo mensagens de erro.