alpha <ed/tech>

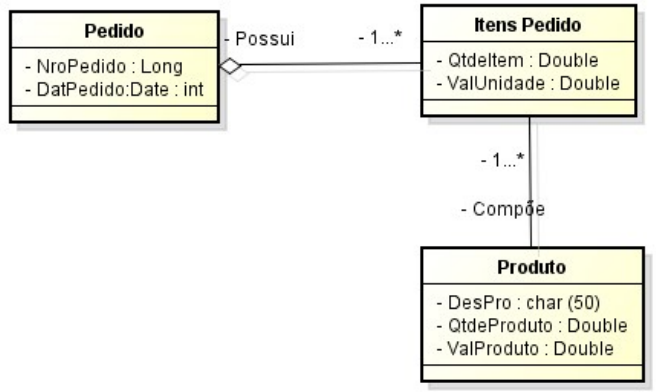
Revisitando Classes

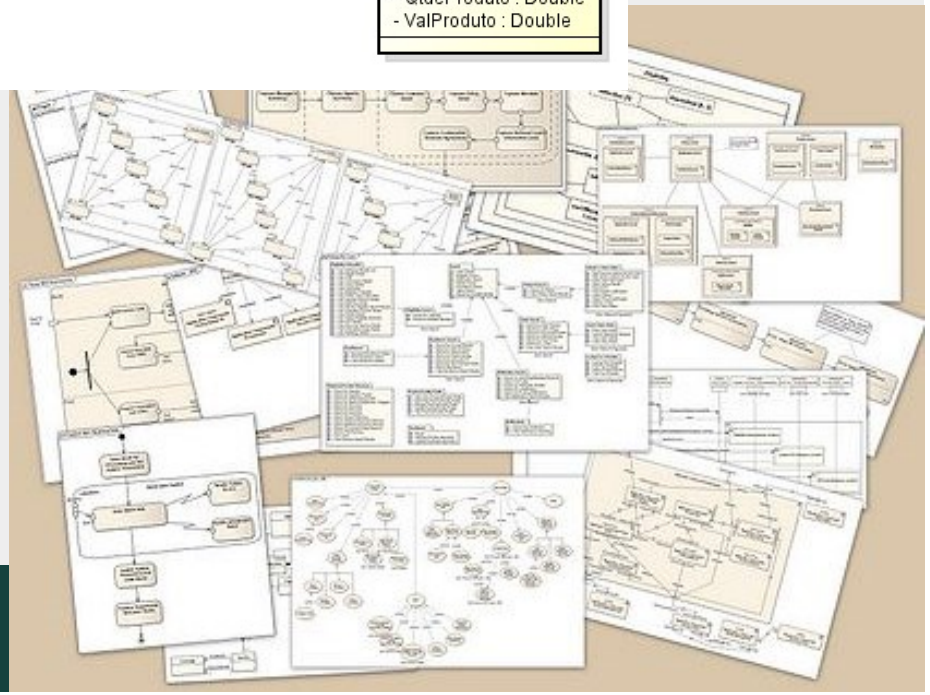
Marcotti

Trilha Python 2024 | Módulo 5 - Aula 6

Classes - OO

- Variáveis são Objetos das Classes
- Classes: int, str, complex, Point, List
- Atributos das Classes
- Métodos das Classes





Classe

- O estado de um objeto representa as coisas que o objeto sabe sobre si mesmo

- Métodos Constructor
- Método __init__
- Decorator
- Métodos Destructor
- Método __del__

Class

- Class – ideia, conjunto dos entes dessa classe
- Constructor __init__
- Instância da Class se torna um Object
- Objeto sabe os valores dos seus atributos
- Destructor __del__
- Atributos
- getters
- setters

Class

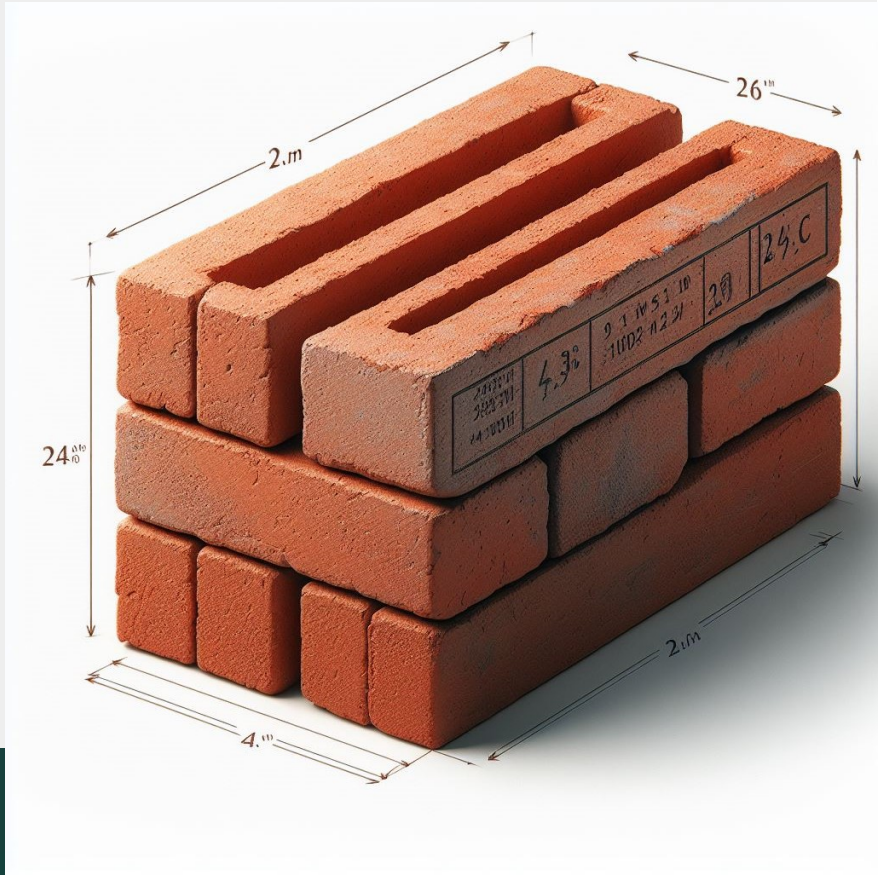
- Class – ideia, conjunto dos entes dessa classe
- Constructor __init__
- Instância da Class se torna um Object
- Object sabe os valores dos seus atributos
- Sabe fazer coisas (métodos)

class Paralelepipedo:

Class

- Class – ideia, conjunto dos entes dessa classe
- Constructor __init__
- Instância da Class se torna um Object
- Objeto sabe os valores dos seus atributos
- Sabe fazer coisas (métodos)

```
class Paralelepipedo:
    def __init__(self, larg, comp, prof):
        self.largura = larg
        self.comprimento = comp
        self.profundidade = prof
```



Atributos

Podemos definir os atributos no início da definição da classe, ou dentro de algum método (normalmente no **constructor __init__**)

Pode existir um método construtor alternativo (usando o decorator **@classmethod**)

```python
class Paralelepipedo:
    def __init__(self, larg, compr, prof):
        self.largura = larg
        self.comprimento = compr
        self.profundidade = prof

    # Implementar método construtor alternativo com decorator
    @classmethod
    def cubo(cls, x=1.0):
        # inicializar um Paralelepipedo com __init__ e os valores x, x, x
        # criando assim um cubo de lado x
        return cls(x, x, x)

    def multiplicar(self, fator=100.0):
        self.largura *= (1.0+fator/100.0)
        self.comprimento *= (1.0+fator/100.0)
        self.profundidade *= (1.0+fator/100.0)
        self.largura = round(self.largura, 2)
        self.comprimento = round(self.comprimento, 2)
        self.profundidade = round(self.profundidade, 2)

    # Implementar método imprimir
    def __str__(self) -> str:
        return f"forma= {self.largura} x {self.comprimento} x {self.profundidade}"
```

Class – Objeto (instância)

```
class Paralelepipedo:
    def __init__(self, larg, comp, prof):
        self.largura = larg
        self.comprimento = comp
        self.profundidade = prof

tijoloBarro = Paralelepipedo(0.15, 0.10, 0.25)
```

Paralelepipedo
-largura: float - comprimento: float - profundidade: float

Instanciando um objeto (“variável”) com nome tijoloBarro1, com valores de atributos definidos

tijoloBarro1 = Paralelepipedo(0.15, 0.25, 0.10)

Class – Objeto (instância)

Paralelepipedo
-largura: float - comprimento: float - profundidade: float
+ multiplicar(fator: float): Paralelepipedo

: Paralelepipedo
largura=0.15 comprimento = 0.25 profundidade = 0.10

tijoloBarro1: Paralelepipedo
largura=0.15 comprimento = 0.25 profundidade = 0.10

Instanciando um objeto (“variável”) com nome tijoloBarro1, com valores de atributos definidos

tijoloBarro1 = Paralelepipedo(0.15, 0.25, 0.10)

Quando instancia um objeto de uma Classe, inicializa valores para cada um dos atributos do objeto

Constructor

- Assinatura do método?
- Constructor __init__

- Atributos
- getters
- setters

```python
class Point:
    """ Point classe para representar e mani
    def __init__(self, initX=0, initY=0):
        self.x = initX
        self.y = initY

    def getX(self):
        return self.x

    def getY(self):
        return self.y


class Turtle:
```

```python
    def andar(self, steps=1) -> None:
        self.position = Point( self.position.getX() + steps * math.sin(self.direction),
                               self.position.getY() + steps * math.cos(self.direction) )

```

```python
    direction: float

    # Implementar método construtor
```

alpha
<ed/tech>

Destructor

- Destructor __del__

```python
class Paralelepipedo:
    qtdParalelepipedo = 0 # Atributo da Classe e não de cada 

    def __init__(self, largura: float = 0.15, comprimento: fl
        self.largura = largura # Atributo de Instância
        self.comprimento = comprimento
        self.profundidade = profundidade
        Paralelepipedo.qtdParalelepipedo += 1

    def __del__(self):
        Paralelepipedo.qtdParalelepipedo -= 1


tijolo1 = Paralelepipedo(0.15, 0.25, 0.10)
tijolo2 = Paralelepipedo(0.35, 0.45, 0.20)

tijolo1.__del__()
print("Quantidade: ", Paralelepipedo.qtdParalelepipedo)

print(tijolo1.__dict__)
```

alpha
<ed/tech>