



universidade de aveiro  
escola superior de tecnologia e  
gestão de água

# Introdução à Programação

Classes

# Programação Orientada a Objetos

Tipos definidos pelo programador para organizar código e dados.

Uma classe é um modelo para criar objetos (uma estrutura organizacional).

O objectivo é representar entidades do mundo real com propriedades (atributos) e comportamentos (métodos).

A um tipo definido pelo programador chamamos de classe.

A definição de classe tem o seguinte aspeto:

```
class Pessoa:
```

```
    """Representa uma pessoa."""
```

O cabeçalho indica que a nova classe se chama Pessoa.

O corpo é um comentário que explica para que serve a classe.

# Programação Orientada a Objetos

Dentro da classe podemos definir variáveis e métodos.

Definir uma classe chamada Pessoa cria um objeto de classe.

O objeto de classe é como uma fábrica para criar objetos.

Para criar uma Pessoa, chamamos Pessoa como se fosse uma função.

```
>>> af = Pessoa()
```

O valor de retorno é uma referência a um objeto Pessoa, que atribuímos a af.

# Programação Orientada a Objetos

A criação de um novo objeto é chamada de **instanciação** e o objeto é uma **instância** da classe.

Você pode atribuir valores a uma instância usando a notação de ponto:

```
>>> af.nome = 'Augusto'
```

```
>>> af.idade = 15
```

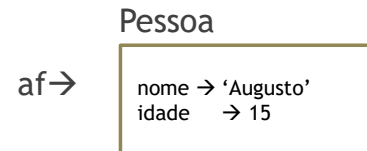
Essa sintaxe é semelhante à sintaxe para selecionar uma variável de um módulo, como `math.pi`.

Neste caso estamos a atribuir valores a elementos do objeto.

A esses elementos damos o nome de **atributos**.

# Diagrama de estado

A figura seguinte é um diagrama de estado e mostra o resultado das atribuições.



Um diagrama de estado que mostra um objeto e seus atributos é chamado de diagrama de objeto

A variável af refere-se a um objeto Pessoa, que contém dois atributos.

Um atributo refere-se a uma string e o outro a um numero inteiro.

Podemos ler o valor de um atributo usando a mesma sintaxe:

```
>>> af.nome
```

```
'Augusto'
```

```
>>> x = af.idade
```

```
>>> x
```

```
15
```

# Atributos

A expressão `af.idade` significa:

“Vai ao objeto ao qual `af` se refere e obtém o valor de `idade`”.

Podemos usar a notação de `af` como parte de qualquer expressão.

```
>>> maioridade = 18 - af.idade
```

```
>>> maioridade
```

```
3
```

Também podemos passar uma instância como um argumento de uma função.

```
def print_pessoa(p):
```

```
    print(f“ O {p.nome} tem {p.idade} anos”)
```

```
>>> print_pessoa(af)
```

```
“O Augusto tem 15 anos”
```

# Sintaxe Básica

```
class NomeDaClasse:  
    """Descrição da Classe"""  
    def __init__(self, parametro1, parametro2):  
        self.atributo1 = parametro1  
        self.atributo2 = parametro2
```

O método `init` (abreviação de “inicialização”) é um método especial que é invocado quando um objeto é instanciado. É utilizada para atribuir valores aos atributos

Seu nome completo é `__init__` (dois caracteres underscore , seguidos por `init` e mais dois underscore ). O método `__init__()`

# Sintaxe Básica - exemplo

Ex:

```
class Pessoa:
```

```
    """Representa uma Pessoa"""
```

```
    def __init__(self, nome, idade):
```

```
        self.nome = nome
```

```
        self.idade = idade
```

```
af= Pessoa("Augusto",15)
```

```
>>> af.nome
```

```
'Augusto'
```

```
>>> x = af.idade
```

```
>>> x
```

```
15
```

# Métodos

Métodos são funções definidas dentro de uma classe que descrevem os comportamentos dos objetos.

```
class Pessoa:
    """Representa uma Pessoa"""
    def __init__(self, nome, idade):
        self.nome = nome
        self.idade = idade

    def apresentar(self):
        print(f'Sou o {self.nome} e tenho {self.idade} anos')
```

```
af= Pessoa('Augusto',15)
af.apresentar()
```

```
>>> Sou o Augusto e tenho 15 anos
```

# Objetos são mutáveis

```
class Pessoa:
    """Representa uma Pessoa"""
    def __init__(self, nome, idade):
        self.nome = nome
        self.idade = idade

    def apresentar(self):
        print(f'Sou o {self.nome} e tenho {self.idade} anos')

    def envelhecer(self, anos):
        self.idade = self.idade + anos
```

← Atua sobre o atributo idade

```
af= Pessoa('Augusto',15)
af.apresentar()

>>> Sou o Augusto e tenho 15 anos
```

```
af.envelhecer(7)
af.apresentar()

>>> Sou o Augusto e tenho 22 anos
```