

Profa. Dra. Raquel C. de Melo-Minardi
Departamento de Ciência da Computação
Instituto de Ciências Exatas
Universidade Federal de Minas Gerais



MÓDULO 2 – PROGRAMAÇÃO

Classes e objetos

ORIENTAÇÃO POR OBJETOS

- ▶ Atenção: esta não é uma aula de **orientação por objetos**
 - ▶ Há inúmeros fundamentos teóricos que não serão abordados aqui
 - ▶ Programação modular
 - ▶ Herança
 - ▶ Encapsulamento
 - ▶ Polimorfismo
 - ▶ Entre outros...
- ▶ Nessa aula, trataremos **apenas da sintaxe em Python** para criação de **classes** e **objetos**

CLASSES E OBJETOS EM PYTHON

- ▶ Python é uma linguagem de programação **orientada por objetos**
 - ▶ Quase tudo em Python é um objeto, com suas propriedades e métodos
 - ▶ Uma **classe** é como um construtor de objetos ou um molde para criar objetos

CRIANDO UMA CLASSE

- ▶ Para criar uma classe chamada MinhaClasse, com uma propriedade chamada x:

```
class MinhaClasse:  
    x = 5
```

CRIANDO UM OBJETO

- ▶ Agora podemos usar a classe chamada MinhaClasse para criar objetos:

```
obj1 = MinhaClasse()  
print(obj1.x)
```

A FUNÇÃO CONSTRUTORA

- ▶ Para entender o significado das classes, precisamos entender a função construtora `__init__()`
- ▶ Todas as classes têm uma função chamada `__init__()`, que é sempre executada quando a classe está sendo iniciada
- ▶ Use a função `__init__()` para atribuir valores a propriedades do objeto ou outras operações necessárias ao objeto que está sendo criado:

```
class Proteina:
    def __init__(self, nome, tamanho):
        self.nome = nome
        self.tamanho = tamanho

p1 = Proteina('Hemoglobina Humana', 252)
print(p1.nome)
```

MÉTODOS DOS OBJETOS

- ▶ Objetos também podem conter métodos
- ▶ Métodos em objetos são funções que pertencem ao objeto
- ▶ Vamos criar um método que imprime uma mensagem na classe Proteína:

```
class Proteína:  
    def __init__(self, nome, tamanho):  
        self.nome = nome  
        self.tamanho = tamanho  
  
    def minhaFuncao(self):  
        print( 'Proteína: ' + self.nome)  
  
p1 = Proteína('Hemoglobina Humana', 252)  
p1.minhaFuncao()
```

- ▶ O parâmetro `self` é uma referência à instância atual da classe e é usado para acessar variáveis que pertencem à classe

O PARÂMETRO SELF

- ▶ O parâmetro `self` é uma referência à instância atual da classe e é usado para acessar variáveis que pertencem à classe
- ▶ Ele não precisa ser nomeado `self`, você pode chamá-lo como quiser, mas tem que ser o primeiro parâmetro de qualquer função na classe:

```
class Proteina:
    def __init__(eu, nome, tamanho):
        eu.nome = nome
        eu.tamanho = tamanho

    def minhaFuncao(euMesma):
        print( 'Proteína: ' + euMesma.nome)

p1 = Proteina('Hemoglobina Humana', 252)
p1.minhaFuncao()
```


MODIFICANDO PARÂMETROS DOS OBJETOS

- ▶ Você pode modificar propriedades em objetos assim:

```
p1.tamanho = 255
```

DELETANDO PARÂMETROS DE OBJETOS

- ▶ Você pode deletar propriedades em objetos usando a palavra chave `del`:

```
del p1.tamanho
```

DELETANDO OBJETOS

- ▶ Você também pode deletar os objetos inteiros usando a palavra chave `del`:

```
del p1
```