



# Linguagem de Programação

Professor Anderson I. S. Abreu

# Funções em Python

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Tópicos da nossa aula:

1. Funções Built-in em Python;
2. Função definida pelo usuário (com retorno e parâmetro);
3. Funções anônimas em Python.

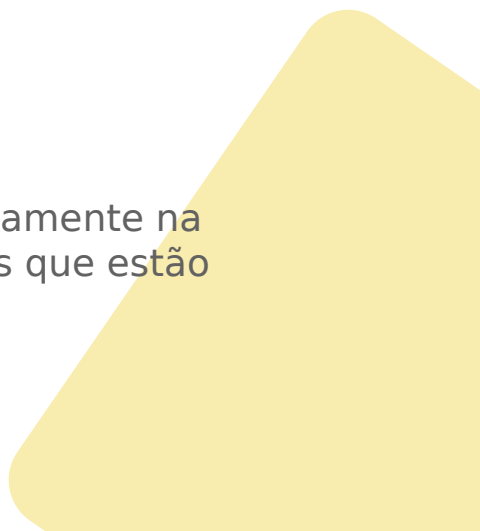


1

# Funções Built-in em Python

---

São blocos de código pré-implementados e incorporados diretamente na linguagem, constituindo um conjunto essencial de ferramentas que estão prontamente disponíveis para os programadores



# Hello World, a função!

Desde que iniciamos nossa jornada na programação com a simples linha de código `print("hello world")`, já estávamos explorando funções;

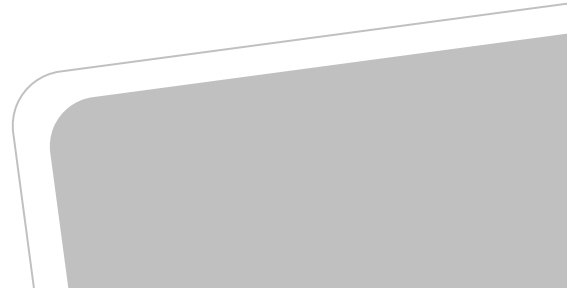
No contexto do Python, pense em funções como ferramentas prontas para uso;

Oferece uma ampla variedade de funções embutidas.

**`print();`**

**`int();`**

**`range().`**



# A função len()

Calcular o comprimento de uma lista e, em seguida, imprimir o resultado com comentários explicativos:

```
# Cria uma lista de números
numeros = [1, 2, 3, 4, 5]

# Usa a função len() para calcular o comprimento da lista
comprimento = len(numeros)

# Imprime o comprimento da lista
print("O comprimento da lista é:", comprimento)
```

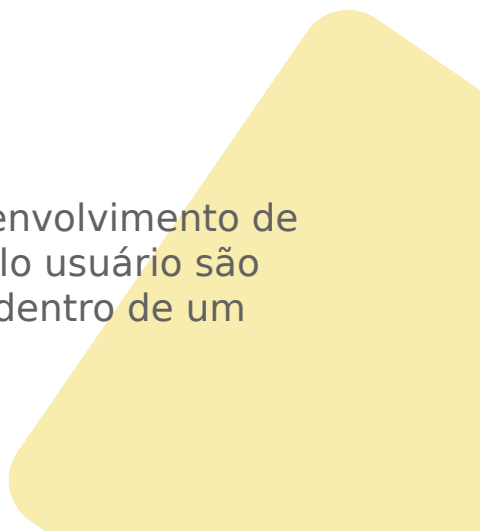


2

## Função definida pelo usuário (com retorno e parâmetro)

---

Representam uma poderosa capacidade de personalização no desenvolvimento de código. Ao contrário das funções built-in, as funções definidas pelo usuário são criadas pelo programador para atender a requisitos específicos dentro de um programa.

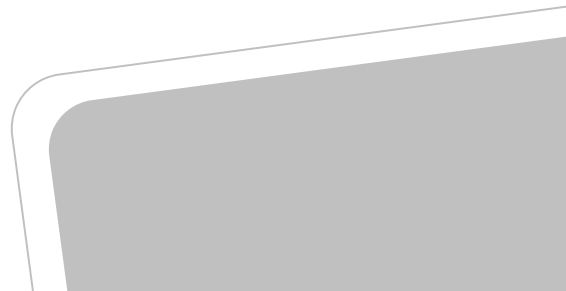


# Nem sempre o que vem pronto funciona!

Cada problema é singular e frequentemente requer abordagens específicas;

Essas funções são pedaços de código que executam ações definidas por nós, os desenvolvedores;

Temos o controle sobre o nome da função, os dados que ela recebe e o resultado que produz.



# Vamos usar a imaginação!

Função  
Soma;

```
# Definindo uma função chamada "soma"
def soma(a, b):
    resultado = a + b
    return resultado
```

Função  
par;      é

```
# Definindo uma função chamada "e_par"
def e_par(numero):
    if numero % 2 == 0:
        return True
    else:
        return False
```





3

# Funções anônimas em Python

---

Conhecidas como funções lambda, oferecem uma abordagem concisa e flexível para a definição de funções pelo usuário. Funções anônimas são declaradas sem a necessidade de um nome, permitindo que sejam definidas e utilizadas no local em que são necessárias.

# Expressão Lambda

As expressões lambda são usadas para criar funções anônimas, o que significa que elas não têm um nome definido com "def.";

Essas funções são úteis quando você precisa de uma ação simples que será usada apenas uma vez;

Ao oferecer essa forma mais compacta de expressar lógica de programação, as funções anônimas enriquecem a expressividade do Python, permitindo que os desenvolvedores criem código mais claro e eficiente em determinados contextos.



# Exemplo anônimo!

```
soma = lambda a, b: a + b  
resultado = soma(3, 4)  
print(resultado)
```

Neste exemplo, criamos uma expressão lambda que realiza a adição de dois números, a e b.

Não atribuímos um nome à função, mas podemos usá-la como qualquer outra função.






# Aplicando a aula!

---

Sendo professor, necessito avaliar constantemente os estudantes, sendo assim, quero automatizar a média de notas dos alunos.

Sabendo como é calcular a média, vamos dar um upgrade no primeiro código utilizando o conhecimento dessa nossa unidade.



# Obrigado!

