

Resumo – Funções

[Definindo funções](#)

[Função que não retorna valor nenhum](#)

[Função com múltiplos parâmetros](#)

[Argumento vs Parâmetro](#)

[Argumentos com valor padrão](#)

Definindo funções

A sintaxe para definir uma função no Python é a seguinte.

```
# Essa é uma função muito simples: recebe um valor e retorna o próprio valor.  
# equivalente a  $f(x) = x$  na matemática  
def f(x):  
    return x  
  
print(f(10)) # Imprime 10 na tela
```

Função que não retorna valor nenhum

A função abaixo apenas imprime valores na tela e “não retorna valor algum”.

```
def imprime_1_2_3():  
    print(1)  
    print(2)  
    print(3)  
  
imprime_1_2_3() # => 1 2 3
```



Apesar de falarmos que ela “não retorna nada”, no Python, toda função tem um `return None` implícito.

```
x = imprime_1_2_3()  
print(x)
```

```
# => None
```

Função com múltiplos parâmetros

Podemos ter funções com múltiplos parâmetros, ou seja, que recebem múltiplos argumentos.

```
def soma(a, b):  
    return a + b
```

Argumento vs Parâmetro



Apesar de muitas vezes utilizarmos os termos “argumento” e “parâmetro” de maneira equivalente, formalmente “parâmetro” se refere às variáveis na assinatura de uma função e argumento aos valores passados quando a função é executada.

```
def soma(a, b): # Parâmetros: a, b  
    return a + b  
  
soma(2, 3) # Argumentos: 2, 3  
soma(4, 5) # Argumentos: 4, 5
```

Argumentos com valor padrão

Documentação oficial: <https://docs.python.org/pt-br/3/tutorial/controlflow.html#default-argument-values>

Podemos definir funções que podem ser invocadas/chamadas com um número variável de argumentos. Uma das maneiras de fazer isso é utilizando argumentos com valor padrão, da seguinte maneira:

```
def calcula_desconto(valor_total, desconto=0.1):  
    """  
    Função que aplica um desconto de 10% no valor_total  
    a não ser que o argumento desconto seja passado.  
    """  
    return valor_total * (1 - desconto)  
  
# Podemos invocar essa função de diferentes maneiras  
calcula_desconto(100, 0.5) # 0.5 substitui o valor padrão 0.1  
=> 50  
  
calcula_desconto(100) # nesse caso, o valor padrão de 0.1 vai ser utilizado  
=> 90
```