



PYTHON
ACADEMY

OPERADORES ARITMÉTICOS E OPERADORES LÓGICOS EM PYTHON

Guia de operadores: aritméticos (+, -, *, /, //, %, **), comparação (==, !=, >, <), lógicos (and, or, not), atribuição (+=, -=), casos práticos.

Este ebook foi gerado por



Crie Ebooks técnicos incríveis em minutos com IA

Conheça a 1ª IA Especializada na criação de Ebooks **com código!**



Chega de formatar código no Google Docs

Deixe que nossa IA faça o trabalho pesado

 Syntax Highlight

 Adicione Banners Promocionais

 Edite em Markdown em Tempo Real

 Infográficos feitos por IA

TESTE AGORA 

 PRIMEIRO CAPÍTULO 100% GRÁTIS

Salve, salve **Pythonista!**

Nesse post vamos falar de **Operadores em Python**: Operadores Lógicos, Operadores Aritméticos, Operadores de Associação, Operadores de Comparação, Operadores de Identidade e Operadores de Atribuição!

Wow, é muito Operador! 🥰

Se você já programa em Python, já se deparou com esses operadores em algum momento: `is`, `is not`, `==`, `in`, `not in`, `and`, `or` ou `%=`.

Nesse post, você vai dominar **TODOS** esses Operadores e aprender como utilizá-los em seus programas Python!

Está pronto, então vamos aprender sobre **Operadores!**

Operadores em Python

Os Operadores em Python possibilitam que o desenvolvedor consiga transcrever a **lógica** para **código**.

Python disponibiliza uma série desses operadores para os desenvolvedores e é muito importante dominá-los se você quiser se tornar um **verdadeiro Pythonista!**

Veremos todos em detalhes agora, começando pelos **Operadores Aritméticos!**

Operadores Aritméticos

Esses operadores são utilizados para criarmos expressões matemáticas comuns, como soma, subtração, multiplicação e divisão.

Veja quais estão disponíveis no Python:

Opera- dor	Nome	Função
+	Adição	Realiza a soma de ambos operandos.
-	Subtração	Realiza a subtração de ambos operandos.
*	Multiplicação	Realiza a multiplicação de ambos operandos.
/	Divisão	Realiza a Divisão de ambos operandos.
//	Divisão inteira	Realiza a divisão entre operandos e a parte decimal de ambos operandos.
%	Módulo	Retorna o resto da divisão de ambos operandos.
**	Exponenciação	Retorna o resultado da elevação da potência pelo outro.

Veja agora a utilização de cada operador aritmético mencionado acima:

```
quatro = 4
dois = 2

soma = quatro + dois
print(soma)  # Resultado: 6

subtracao = quatro - dois
print(subtracao)  # Resultado: 2

multiplicacao = quatro * dois
print(multiplicacao)  # Resultado: 8

divisao = quatro / dois
print(divisao)  # Resultado: 2.0

divisao_interna = quatro // dois
print(divisao_interna)  # Resultado: 2

modulo = quatro % dois
print(modulo)  # Resultado: 0

exponenciacao = quatro ** dois
print(exponenciacao)  # Resultado: 16
```

Operadores de Comparação

Como o nome já diz, esses Operadores são usados para **comparar** dois valores:

Opera- dor	Nome	Função
<code>==</code>	Igual a	Verifica se um valor é igual ao outro
<code>!=</code>	Diferente de	Verifica se um valor é diferente ao outro
<code>></code>	Maior que	Verifica se um valor é maior que outro
<code>>=</code>	Maior ou igual	Verifica se um valor é maior ou igual ao outro
<code><</code>	Menor que	Verifica se um valor é menor que outro
<code><=</code>	Menor ou igual	Verifica se um valor é menor ou igual ao outro

Vamos ver exemplos da utilização de cada operador de comparação mencionado acima.

Para facilitar o entendimento, todas as operações estão retornando um valor igual a `True`, para que você entenda como cada condição é aceita:


```
var = 5

if var == 5:
    print('Os valores são iguais')

if var != 7:
    print('O valor não é igual a 7')

if var > 2:
    print('O valor da variável é maior de 2')

if var >= 5:
    print('O valor da variável é maior ou igual a 5')

if var < 7:
    print('O valor da variável é menor que 7')

if var <= 5:
    print('O valor da variável é menor ou igual a 5')
```

Resultado do código acima:

```
Os valores são iguais
O valor não é igual a 5
O valor da variável é maior de 5
O valor da variável é maior ou igual a 5
O valor da variável é menor que 7
O valor da variável é menor ou igual a 5
```



*Estou desenvolvendo o **DevBook**, uma plataforma que usa IA para gerar ebooks técnicos profissionais. Depois de ler, dá uma passada no site!*

Crie Ebooks técnicos incríveis em minutos com IA

Conheça a 1ª IA Especializada na criação de Ebooks **com código!**



Chega de formatar código no Google Docs

Deixe que nossa IA faça o trabalho pesado

Syntax Highlight

Adicione Banners Promocionais

Edite em Markdown em Tempo Real

Infográficos feitos por IA

TESTE AGORA! PRIMEIRO CAPÍTULO 100% GRÁTIS

Operadores de Atribuição

Esses Operadores são utilizados no momento da **atribuição** de valores à variáveis e controlam como a atribuição será realizada.

Veja quais Operadores de Atribuição estão disponíveis em Python:

Operador	Equivalente a
<code>=</code>	$x = 1$
<code>+=</code>	$x = x + 1$
<code>-=</code>	$x = x - 1$
<code>*=</code>	$x = x * 1$
<code>/=</code>	$x = x / 1$
<code>%=</code>	$x = x \% 1$

Exemplo da utilização de cada operador de atribuição mencionado acima:

👉 Operador `+=` :

```
numero = 5

numero += 7
print(numero) # Resultado será 10
```

👉 Operador `-=` :

```
numero = 5

numero -= 3
print(numero) # Resultado será 2
```

👉 Operador `*=` :

```
numero = 5

numero *= 2
print(numero)  # Resultado será 10
```

👉 Operador `/=` :

```
numero = 5

numero /= 4
print(numero)  # Resultado será 1.25
```

👉 Operador `%=` :

```
numero = 5

numero %= 2
print(numero)  # Resultado será 1
```

Obs: O operador `%` é chamado módulo e nada mais é que o resto da divisão. No exemplo acima: 5 dividido por 2 dá 2 de resultado e sobra 1. Por isso `numero %= 2` será `1` !

Operadores Lógicos

Esses Operadores nos possibilitam construir um tipo de teste muito útil e muito utilizado em qualquer programa Python: os **testes lógicos**.

Python nos disponibiliza três tipos de Operadores Lógicos: o `and`, o `or` e o `not`.

Vamos ver mais sobre eles agora!

Ope- rador	Definição
<code>and</code>	Retorna True se ambas as afirmações forem verdadeiras
<code>or</code>	Retorna True se uma das afirmações for verdadeira
<code>not</code>	retorna Falso se o resultado for verdadeiro

Exemplo da utilização de cada um:

```
num1 = 7
num2 = 4

# Exemplo and
if num1 > 3 and num2 < 8:
    print("As Duas condições são verdadeiras")

# Exemplo or
if num1 > 4 or num2 <= 8:
    print("Uma ou duas das condições são verdadeiras")

# Exemplo not
if not (num1 < 30 and num2 < 8):
    print('Inverte o resultado da condição entre os parênteses')
```

Operadores de Identidade

Estes Operadores são utilizados para *comparar* objetos, verificando se os objetos testados referenciam o mesmo objeto (`is`) ou não (`is not`).

Ope- rador	Definição
<code>is</code>	Retorna <code>True</code> se ambas as variáveis são o mesmo objeto
<code>is</code> <code>not</code>	Retorna <code>True</code> se ambas as variáveis não forem o mesmo objeto

Agora vamos aos exemplos de como utilizar cada operador de identidade mencionado acima:

Exemplo do operador `is`:

```
lista = [1, 2, 3]
outra_lista = [1, 2, 3]
recebe_lista = lista

# Recebe True, pois são o mesmo objeto
print(f"São o mesmo objeto? {lista is recebe_lista}")

# Retorna False, pois são objetos diferentes
print(f"São o mesmo objeto? {lista is outra_lista}")
```

Resultado do código acima:

```
São o mesmo objeto? True
São o mesmo objeto? False
```

Exemplo do operador `is not`:

```
tupla = (1, 2, 3)
outra_tupla = (1, 2, 3)

print(f"Os objetos são diferentes? {outra_tupla is tupla}")
```

Resultado do código acima:

```
Os objetos são diferentes? True
```

Muitas vezes programadores Python ficam na dúvida em quando utilizar o operador de igualdade `==` ou o operador de identidade `is`.

Mas agora que você já conhece os dois sabe que o operador `==` verifica os **valores** testados, enquanto o operador `is` testa a **referência** dos valores testados!



Operadores de Associação

Por último, temos os Operadores de Associação.

Eles servem para verificar se determinado objeto está **associado** ou **pertence** a determinada estrutura de dados.

Ope- rador	Função
<code>in</code>	Retorna <code>True</code> caso o valor seja encontrado na sequência
<code>not in</code>	Retorna <code>True</code> caso o valor não seja encontrado na sequência

Exemplo da utilização de cada operador de associação mencionado acima:

```
lista = ["Python", 'Academy', "Operadores", 'Condições']

# Verifica se existe a string dentro da lista
print('Python' in lista) # Saída: True

# Verifica se não existe a string dentro da lista
print('SQL' not in lista) # Saída: True
```

Conclusão

Vimos nesse post os diversos tipos de Operadores que o Python disponibiliza para que possamos desenvolver nossos scripts na Linguagem!

Agora que você já sabe como utilizá-los, você deu um passo importante para conhecer a fundo a linguagem Python!

Se ficou com alguma dúvida, fique à vontade para deixar um comentário no box aqui embaixo! Será um prazer te responder! 😊

Não se esqueça de conferir!



DevBook

Crie Ebooks técnicos em minutos com IA

Conheça a 1ª IA Especializada na criação de Ebooks **com código!**



Chega de formatar código no Google Docs



 Syntax Highlight

 Infográficos feitos por IA

 Adicione Banners Promocionais

Deixe que nossa IA faça o trabalho pesado

 Edite em Markdown em Tempo Real

TESTE AGORA 

 PRIMEIRO CAPÍTULO 100% GRÁTIS