

## OPERAÇÕES BÁSICAS

# OPERAÇÕES BÁSICAS COM O PHYTON

Neste vídeo, [Marcus Oliveira](#), Gerente de Data Science do Quintoandar, apresenta as operações básicas com o Python.



04:01



## OPERAÇÕES MATEMÁTICAS

Podemos começar a brincar com o Python pensando nele como uma poderosa calculadora.

Para isso, vamos aprender a executar as principais operações matemáticas, como ele entende cada tipo de dado que inserimos e como guardar essas informações dentro de variáveis para que possamos recuperá-las depois.

- Soma: **1 + 1**
- Subtração: **1 - 1**
- Multiplicação: **1 \* 1**
- Divisão: **1 / 1**

## OPERAÇÕES BÁSICAS

- Exponenciação ou potenciação:  $1^{**} 1$
  - Tirar a raiz (que é um tipo de exponenciação):  $1^{**} (-1/2)$
  - Obter apenas o número inteiro da divisão:  $1 // 1$
  - Obter o resto da divisão:  $1 \% 1$
- 

AVANÇAR

## OPERAÇÕES BÁSICAS

### VARIÁVEIS

Marcus apresenta agora como funcionam e para que servem as *variáveis* no Python.



05:18



### TIPOS DE DADOS MAIS COMUNS

Ao longo de sua jornada em Python, é provável que você encontre pelo menos quatro tipo de dados:

- *String*: letras ou texto (e.g. 'A' ou 'apple')
- *Integer*: número inteiro (e.g. 1 ou 123)
- *Float*: número decimal (e.g. 1.0 ou 1.42)
- *Boolean*: verdadeiro ou falso (i.e. 'True' ou 'False')

### POR QUE ENTENDER CADA TIPO DE DADO?

## OPERAÇÕES BÁSICAS

aumenta pois também será necessário entender como melhor utilizar cada tipo de variável empregada.

Por exemplo, uma variável *string* 'maçã' não é um valor numérico, porém o Python aceita aplicar a operação de multiplicação. Caso multipliquemos a variável 'maçã' por 2, teremos 'maçãmaçã'. Ou seja, o Python repetiu a variável duas vezes.

Agora, imagine um segundo cenário, onde a pessoa que está escrevendo o código criou, acidentalmente, uma variável texto que deveria ter sido numérica. Ao aplicar uma operação matemática, um comportamento inesperado pode surgir e tornar-se um problema.

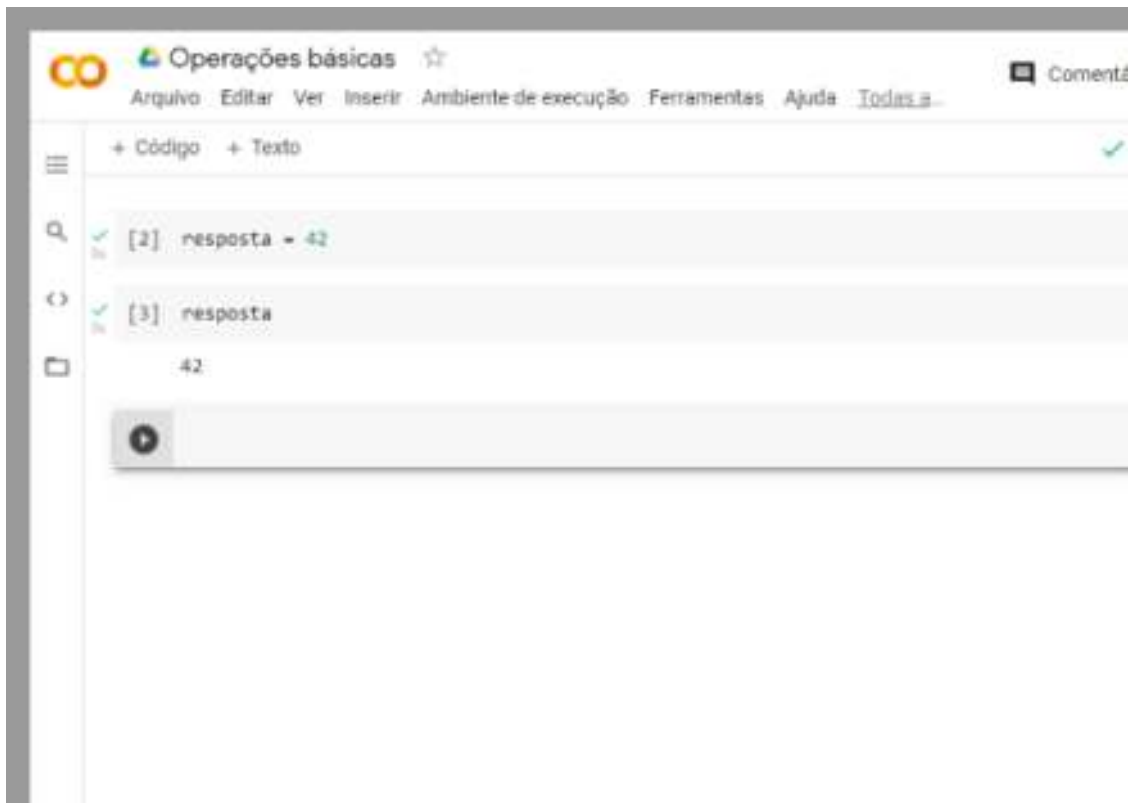
## GUARDANDO VALORES EM VARIÁVEIS

Esse é um dos conceitos mais utilizados nas linguagens de programação. São utilizadas para armazenar valores na memória, tornando possível gravar e ler esses valores em um nome pré-definido.

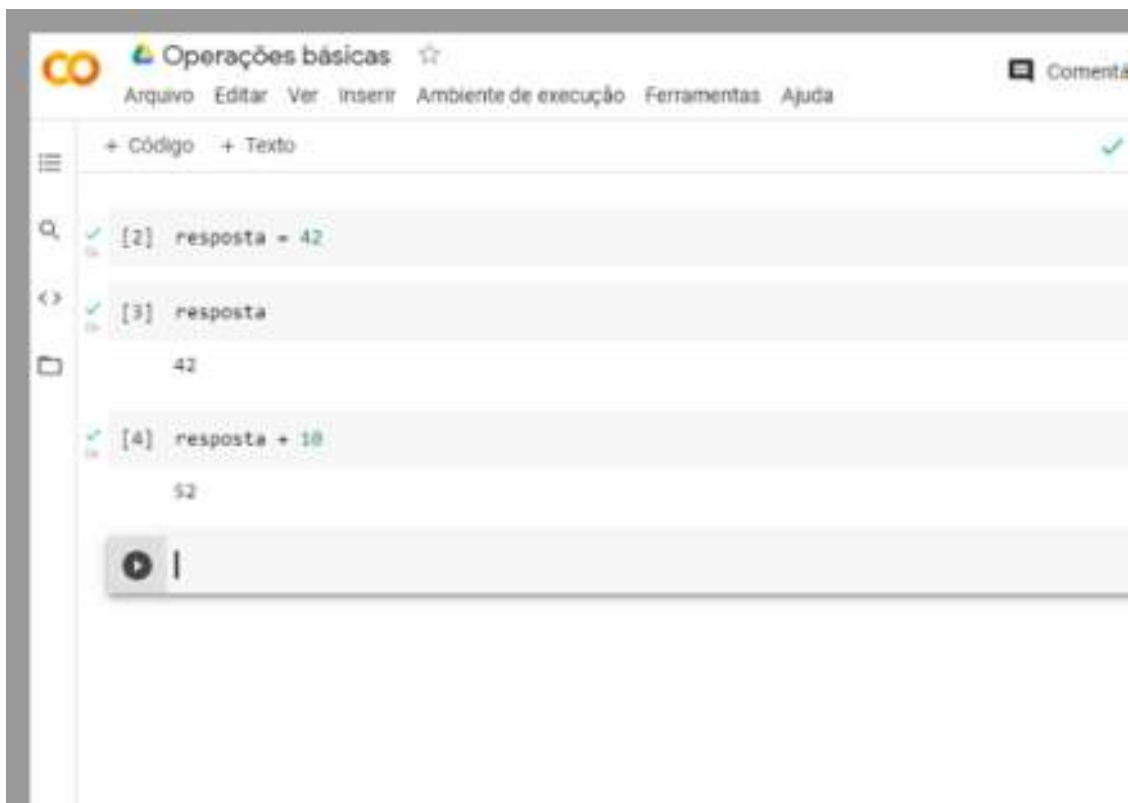
Vamos guardar o número **42** dentro da variável chamada **resposta**:

```
resposta = 42
```

## OPERAÇÕES BÁSICAS

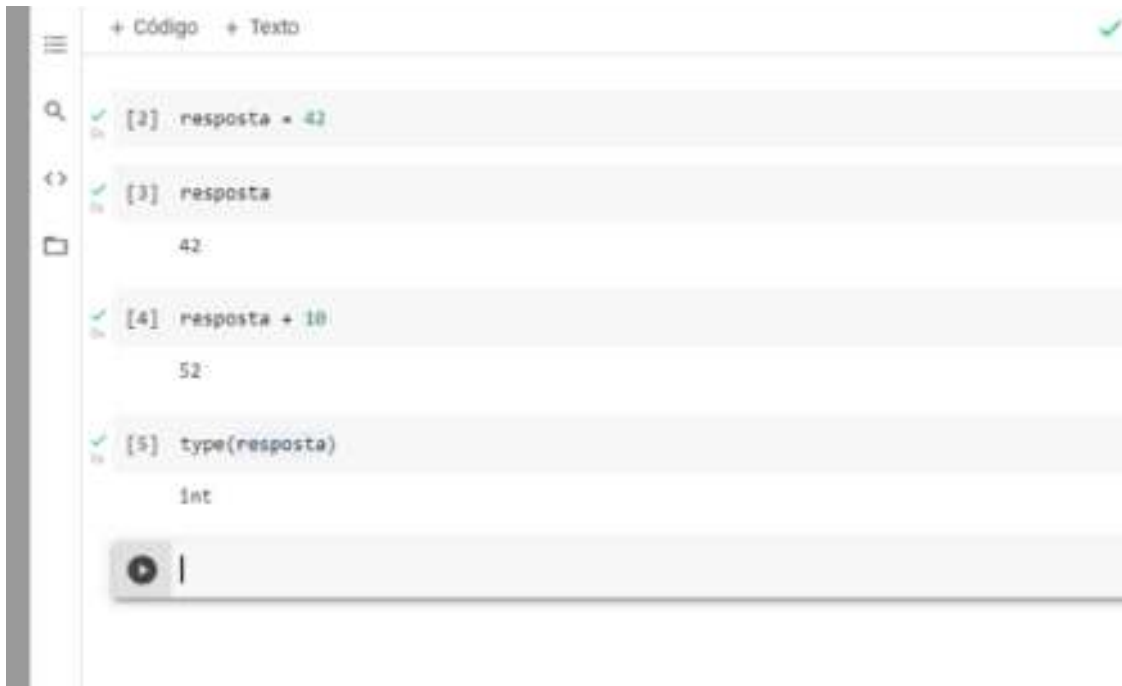


Podemos fazer operações com as variáveis:



Para sabermos o **tipo** de dado que está dentro da variável podemos utilizar a função **type()**

## OPERAÇÕES BÁSICAS



The screenshot shows a Python REPL interface with a sidebar on the left containing icons for a menu, search, code editor, and file explorer. The main area displays a series of code snippets with their execution results:

- Snippet [2]: `resposta = 42` (result: 42)
- Snippet [3]: `resposta` (result: 42)
- Snippet [4]: `resposta + 10` (result: 52)
- Snippet [5]: `type(resposta)` (result: `int`)

At the bottom, there is a play button icon and a vertical bar, indicating the next step in the execution.

Neste caso, nossa variável `resposta` é do tipo **inteiro**.

## AVANÇANDO O PRINT COM AS F-STRINGS

Já vimos como fazer o Python imprimir uma frase (string) e também aprendemos sobre variáveis.

Vamos avançar um pouco mais e ver como podemos tornar essa função um pouco mais dinâmica.

As **F-strings** são uma maneira de incorporar variáveis dentro das strings, para isso, basta colocarmos a letra `f` antes das aspas que iniciam a string e colocar a variável que eu quero que seja incorporada entre chaves.

Vamos ver um exemplo:

```
sobremesa = "pudim"
print(f"Hoje a sobremesa é: {sobremesa}!")
Hoje a sobremesa é: pudim!
```