

## Números

Para utilizares números escreve-os como estás habituado. Se precisares de casas decimais, tens que utilizar um ponto (.) em vez de uma vírgula (,).

Python	comentário
7	# Número inteiro.
-1	# Número inteiro negativo.
2.5	# Número com uma casa decimal.

## Contas

Podes fazer quaisquer contas em Python, mas os símbolos das operações são um pouco diferentes.

Python	resultado	comentário
3 + 5	8	# Adição.
9 - 2	7	# Subtração.
3 * 2	6	# Multiplicação.
8 / 2	4	# Divisão.

Também podes agrupar operações com parêntesis.

Python	resultado
(1 + 2) * 3	9
(4 + 6) / (7 - 2)	2

## Texto

Em Python, chamamos **string** a bocadinhos de texto (palavras ou frases). O texto representa-se sempre entre aspas (") ao longo de uma só linha.

Python	comentário
"Maria"	# Um nome.
"Olá a todos!"	# Uma frase.

Letras sem aspas (") são variáveis.

## Escrever e Fazer Perguntas

A função **print** escreve números ou texto. Usa um ou mais parâmetros separados por vírgulas:

escrever em Python	comentário
print("Olá")	# Escreve <b>Olá</b> .
print(3 + 5)	# Escreve <b>8</b> .
print(7, "maçãs")	# Escreve <b>7 maçãs</b> .

perguntas em Python
# Associa o texto da resposta à variável <b>nome</b> . nome = input("Como te chamas?")
# Para fazer contas, pega-se no texto da resposta # <b>digitos</b> e obtém-se um número com <b>int(digitos)</b> . digitos = input("Qual o número?") numero = int(digitos)

## Variáveis

São utilizadas para que o Python se lembre de coisas. Cada variável tem um nome, com uma ou mais letras, que formam uma palavra, mas sem usar aspas ("). Cada variável pode ser associada a um número, a uma string (texto), a uma tartaruga, ou a outras coisas.

associar valores	comentário
quem = "Maria"	# A variável <b>quem</b> fica # associada à string <b>"Maria"</b> .
idade = 12	# A variável <b>idade</b> fica # associada ao número <b>12</b> .

Ao encontrar uma variável, o Python lembra-se do valor que lhe está associado e utiliza-o no seu lugar.

usar variáveis
# Escreve <b>Maria</b> porque # a variável <b>quem</b> está associada à string <b>"Maria"</b> . print(quem)
# Escreve <b>15</b> porque a variável <b>idade</b> está associada # ao número <b>12</b> que é usado na conta <b>idade + 3</b> . print(idade + 3)
# Escreve <b>Maria tem 12 anos</b> . print(quem, "tem", idade, "anos.")
# Calcula quantos anos faltam até aos 20, e associa # o resultado, que é o número <b>8</b> , à variável <b>faltam</b> . faltam = 20 - idade



## Condições

A instrução `if` executa uma ou mais instruções quando a **condição** à sua frente é verdadeira.

### receita Python: execução condicional

```
if condição:
    # uma ou mais instruções, 4 espaços à direita
```

A condição normalmente compara o valor associado a uma **variável** com um valor específico. Exemplos:

condição	comentário
<code>variável == valor</code>	# Verdade se são iguais.
<code>variável != valor</code>	# Verdade se são diferentes.
<code>variável &lt; valor</code>	# Verdade se a variável é menor.
<code>variável &gt; valor</code>	# Verdade se a variável é maior.

Podes usar condições mais complexas com **and** e **or**:

condição	comentário
<code>condição and condição</code>	# Verdade se ambas as # condições são verdade.
<code>condição or condição</code>	# Verdade se pelo menos # uma condição é verdade.

### exemplo (assumindo que as variáveis existem!)

```
if hora == 7 and minuto == 15:
    print("Acordar!")
    print("Lavar os dentes.")
```

## Ciclos

Os ciclos são formas de executar grupos de uma ou mais instruções repetidamente.

### receita Python: repetir um número de vezes

```
for vez in range(número):
    print(vez)
```

Esta receita faz com que a instrução `print` seja executada tantas vezes quantas o **número** indicado.

Na primeira execução, a variável **vez** é associada ao número **0**, na segunda ao número **1**, e por aí fora.

O exemplo repete uma só instrução, mas poderia repetir várias: todas as que estivessem agrupadas depois da linha do **for**, 4 espaços à direita.

## Valores Aleatórios

São valores obtidos ao calhas durante a execução, como se se tratasse do lançamento de um dado:

### receitas Python

```
# Diz ao Python que vamos usar valores aleatórios.
# Deve estar mesmo no início do programa.
import random
```

```
# Um número ao calhas entre min e max.
random.randint(min, max)
```

```
# Um dos elementos ao calhas, que devem ser nú-
# meros ou strings (texto), separados por vírgulas.
random.choice([elementos])
```

## Funções

São grupos de instruções associados a um nome.

### definir uma função chamada cumprimenta

```
def cumprimenta():
    print("Olá")
    print("Como estás?")
```

Quando definidas, as funções são como uma nova instrução. Para as executar (e às suas instruções), escreve o seu nome seguido de parêntesis.

### usar a função cumprimenta

```
cumprimenta()
```

As funções podem ter parâmetros, que são como variáveis, mas que só existem para as suas instruções.

### a função cumprimenta com um parâmetro nome

```
def cumprimenta(nome):
    print("Olá", nome)
    print("Como estás?")
```

### usar a função cumprimenta com o parâmetro

```
# Vai escrever Olá Maria e depois Como estás?
cumprimenta("Maria")
```

Às vezes, podes querer interromper a execução de uma função a meio. Talvez, conforme uma condição!

Para interromper a execução de uma função antes de chegar ao final, utiliza a instrução **return**.