Funções em Python

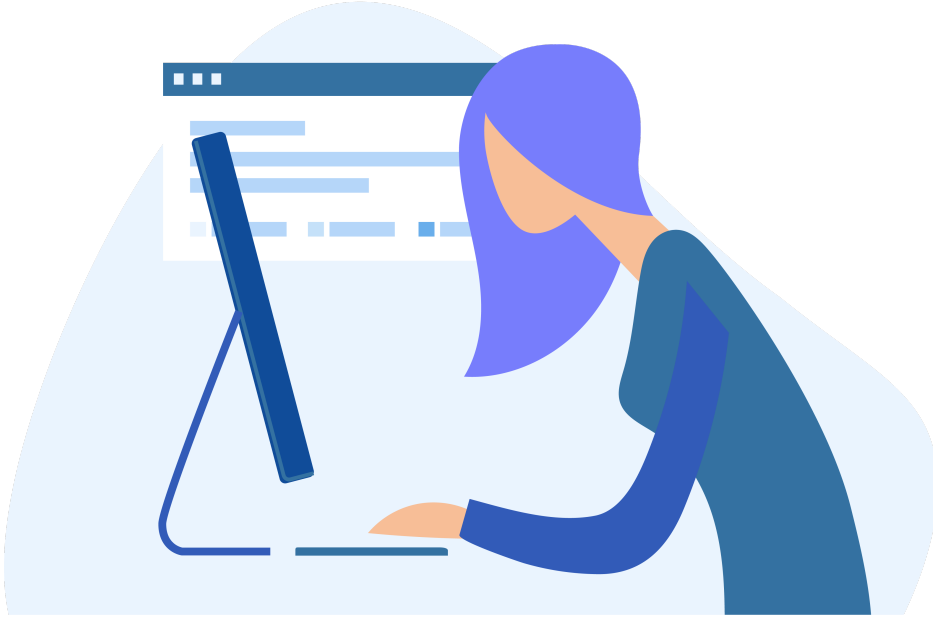
Você sabia que seu material didático é interativo e multimídia? Isso significa que você pode interagir com o conteúdo de diversas formas, a qualquer hora e lugar. Na versão impressa, porém, alguns conteúdos interativos ficam desabilitados. Por essa razão, fique atento: sempre que possível, opte pela versão digital. Bons estudos!

Uma função é uma forma de organização, usada para delimitar ou determinar quais tarefas podem ser realizadas por uma determinada divisão. Em linguagem de programação, uma função é uma sequência de instruções que processa um ou mais resultados que são chamados de parâmetros. Em Python temos as funções *built-in* e as funções definidas pelo usuário.

Funções *built-in* em Python

Uma função *built-in* é um objeto que está integrado ao núcleo do interpretador Python. Ou seja, não precisa ser feita nenhuma instalação adicional, já está pronto para uso.

O interpretador Python possui várias funções disponíveis, conforme podemos conservar na tabela de *Funções built-in* em Python.



Fonte: Shutterstock.

Built-in em Python

abs()	delattr()	hash()	memoryview()	set()
all()	dict()	help()	min()	setattr()
any()	dir()	hex()	next()	slice()
ascii()	divmod()	id()	object()	sorted()
bin()	enumerate()	input()	oct()	staticmethod()
bool()	eval()	int()	open()	str()
breakpoint()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()
complex()	hasattr()	max()	round()	

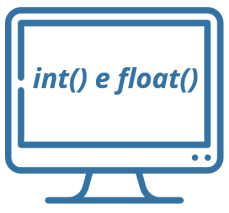
Fonte: Python (2020a).

Veja a descrição de algumas dessas funções:

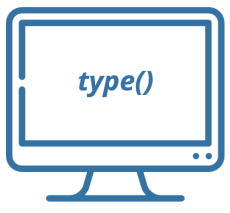
Usada para imprimir um valor na tela.

Usada para retornar a posição de um valor em uma sequência.

Usada para capturar um valor digitado no teclado.



Usadas para converter um valor no tipo inteiro ou float.

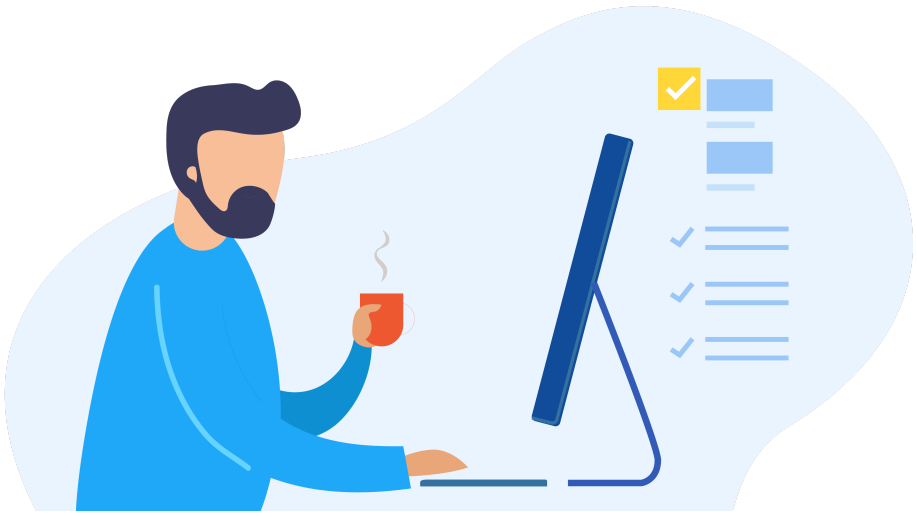


Usada para saber qual é o tipo de um objeto (variável).

Função definidas pelo usuário

Além das Funções *built-in*, muitas vezes as soluções exigem a implementação de funções específicas, as quais são chamadas de funções definidas pelo usuário.

> Funções, também conhecidas como subprogramas ou sub-rotinas, são pequenos blocos de código aos quais se dá um nome, desenvolvidos para resolver tarefas específicas. Tais funções constituem um elemento de fundamental importância na moderna programação de computadores, a ponto de ser possível afirmar que atualmente nenhum programa de computador é desenvolvido sem o uso desse recurso.
>
> — BANIN, Sérgio Luiz. **Python 3 - conceitos e aplicações:** uma abordagem didática. São Paulo: Érica, 2018.



Fonte: Shutterstock.

A sintaxe de uma função em Python é feita com:

» A palavra reservada **def**.
» O nome da função.
» Os parênteses que indicam se existem ou não parâmetros para a função.
» E o comando **return** (que é opcional);

A seguir, veja que existem três sintaxes diferentes:

1 Uma função que não recebe e não retorna valores.

```
1 def nome_função ():
2     # bloco de comandos
```

Sem parâmetros

Identação

Fonte: elaborada pela autora.

2 Uma função que não recebe argumento, mas retorna valores.

```
1 def nome_função ():
2  # bloco de comandos
3     Return valor
```

Sem parâmetros

Com retorno

Fonte: elaborada pela autora.

3 Uma função que recebe argumento e retorna valor.

```
1 def nome_função(param1, param2):
2  # bloco de comandos
3  Return valor
```

Com parâmetros

Com retorno

Fonte: elaborada pela autora.

Funções anônimas em Python

Expressões lambda (às vezes chamadas de formas lambda) são usadas para criar funções anônimas. Uma função anônima é uma função que não é construída com o "***def***" e que, por isso, não possui nome. Esse tipo de construção é útil quando a função faz somente uma ação e é usada uma única vez.

Pesquise mais

Os argumentos de uma função podem ser posicionais ou nominais. No primeiro, cada argumento será acessado pela sua posição; no segundo, pelo nome. Além disso, os parâmetros podem ter valores padrões, o que torna sua passagem não obrigatória.

Leia as seções 5.3.3 (Parâmetros com valores-padrão) e 5.3.4 (Parâmetros nomeados) da obra: BANIN, Sérgio Luiz. **Python 3 - conceitos e aplicações:** uma abordagem didática. São Paulo: Érica, 2018.

Fonte: Shutterstock.