

DICT COMPREHENSIONS NO PYTHON

 Curso de Python

(<https://go.hotmart.com/A55372730Q>) Gostou do conteúdo? Compartilha aí!

Fala grande **Dev!**



(<https://go.hotmart.com/A55372730Q>) Gostou do conteúdo? Compartilha aí!

Dict Comprehensions é uma ferramenta muito útil e poderosa do Python para manipulação de dicionários (dicts)! Com essa técnica, podemos tratar dicionários de uma maneira rápida e concisa!

Essa técnica segue o mesmo conceito das mais conhecidas *List Comprehensions*!

Ainda não sabe manusear listas usando List Comprehensions? 🤔

Então P-A-R-A tudo e vai lá no post sobre List Comprehensions (<https://pythonacademy.com.br/blog/list-comprehensions-no-python>). Fico te aguardando voltar!

... Tô esperando ⌚ ...

Pronto! Agora podemos continuar! 😊

Vá Direto ao Assunto...

- O início de tudo: o dict
- *Dict Comprehensions*
- *Dict Comprehensions* com if
- *Dict Comprehensions* com vários if

O início de tudo: o dict

Gostou do conteúdo? Compartilha aí!

Antes de tudo, vamos falar rapidamente sobre o tipo de dado dict. (Se você já sabe, pode pular

essa seção 😊



(<https://pythonacademy.com.br/blog/list-comprehensions-no-python>)

Dicionário em Python é uma coleção de dados sem ordem onde cada elemento possui um par chave/valor.

E o que é um par chave/valor?

Basicamente é uma forma de se indexar um valor a partir de uma chave.

Isso dá acesso eficiente (pros puristas, temos **O(1)** aqui!) aos valores da estrutura de dados.

A sintaxe para **criar** dicionários é a seguinte:

```
1  # Dicionário vazio
2  dicionario = {}
3
4  # Dicionário comum
5  dicionario = {'jedi': 10, 'sith': 7}
6
7  # Dicionário com chaves inteiras
8  dicionario = {1: 'Baby Yoda', 2: 'Yoda'}
9
10 # Dicionário misturado
11 dicionario = {'especie': 'Humano', 1: ['Obi Wan Kenobi', 'Qui-Gon Jinn']}
12
13 # Outra forma de criação, usando dict()
14 dicionario = dict({'jedi': 10, 'sith': 7})
```

Já para **acessar** elementos:

Gostou do conteúdo? Compartilha aí!



```
1  dicionario = {'nome': 'Vinícius Ramos', 'idade': 29}
2
3  # Saída: Vinícius
4  print(dicionario['nome'])
5
6  # Saída: 29
7  print(dicionario.get('idade'))
8
9  # Caso não encontre, devolva o valor None
10 print(dicionario.get('altura', None))
11
12 # Será Lançada uma exceção KeyError
13 print(dicionario['endereço'])
```

Para **atualizar** valores:

```
1  dicionario = {'nome': 'Vinícius Ramos', 'idade': 29, 'empresa': 'PythonAcademy'}
2
3  # Atualiza dados
4  dicionario['idade'] = 30
5  dicionario['empresa'] = 'PythonAcademy Inc.'
```

Para **remover** elementos:

Gostou do conteúdo? Compartilha aí!



```
1  dicionario = {'nome': 'Vinícius Ramos', 'idade': 29, 'empresa': 'PythonAcademy'}
2
3  # Remove a chave/valor 'idade': 29
4  dicionario.pop('idade')
5
6  # Remove um par aleatório
7  dicionario.popitem()
8
9  # Remove todos os itens
10 dicionario.clear()
11
12 # Deleta 'empresa'
13 del dicionario['empresa']
```

Pronto! Você agora é um expert em Dicionários!

Agora vamos pro prato principal: *Dict Comprehension*!

Antes que eu me esqueça! Se não é inscrito no canal, curta, dê um jóinh.....

Ops! Canal errado! 😊

Mas se está gostando no nosso blog e conteúdo, que tal fazer parte da nossa **lista de emails e não perder nossas atualizações?**

Gostou do conteúdo? Compartilha aí!



Dict Comprehensions

Dict Comprehensions foram introduzidas na linguagem através da especificação PEP 274 (<https://www.python.org/dev/peps/pep-0274/>).

Sua sintaxe básica é:

```
1 {chave: valor for elemento in iteravel}
```

Agora respira que vamos entender cada ponto:

- ◆ `chave` : será a chave de cada elemento do dicionário resultante.
- ◆ `valor` : valor daquela chave.
- ◆ `elemento` : é a unidade de iteração do iterável `iteravel` (se for uma lista, por exemplo, `elemento` irá receber o valor iteração à iteração)
- ◆ `iteravel` : conjunto de dados que estão sendo iterados (pode ser uma lista ou um `set`, por exemplo)

Pra esclarecer, vamos à um exemplo:

Gostou do conteúdo? Compartilha aí!

```
1
```

```
dicionario = {elemento: elemento*2 for elemento in range(6)}
```



(clique aqui para se inscrever via RSS) (clique aqui para se inscrever via Email) (clique aqui para se inscrever via Telegram) (clique aqui para se inscrever via Facebook) (clique aqui para se inscrever via Twitter)

Aqui, cada elemento da lista resultante de `range(6)` (0, 1, 2, 3, 4, 5) será convertido em:

- ◆ Uma chave com o mesmo valor do elemento da lista.
- ◆ `elemento*2` é o valor de cada chave (multiplicar por 2 cada elemento).

O resultado será:

```
{0: 0, 1: 2, 2: 4, 3: 6, 4: 8, 5: 10}
```

Resultado

Outro exemplo, com chaves alfabéticas e manipulação de strings com *f-strings*:

```
1 lista = ['Ferrari', 'Lamborghini', 'Porsche']
2 dicionario = {
3     f'{elemento.lower()}': f'Montadora: {elemento.upper()}' for elemento in lista
4 }
```

Resultando em:

```
{
  'ferrari': 'Montadora: FERRARI',
  'lamborghini': 'Montadora: LAMBORGHINI',
  'porsche': 'Montadora: PORSCHE'
}
```

Resultado

Também é possível iterar sobre um outro dicionário através do método `items()`.

Gostou do conteúdo? Compartilha aí!

Ele retorna a chave e o valor de cada elemento do dicionário de entrada.



Veja um exemplo:

```

1  import locale
2
3  # Configura o Locale pra Português do Brasil (pt_BR)
4  locale.setlocale(locale.LC_MONETARY, 'pt_BR.utf8')
5
6  carros_esportivos = {
7      'ferrari': 1299000,
8      'lamborghini': 1100000,
9      'porsche': 759000
10 }
11
12 dict_saida = {
13     chave: f'{chave.upper()}: {locale.currency(valor)}' for chave, valor in carros_esportivos.items()
14 }
```

Essa seria a saída:

```

{
  'ferrari': 'FERRARI: R$ 1299000,00',
  'lamborghini': 'LAMBORGHINI: R$ 1100000,00',
  'porsche': 'PORSCHE: R$ 759000,00'
}
```

Resultado

Legal não é mesmo?!

Assim como acontece em *List Comprehensions*, também podemos adicionar lógica condicional

(*if* / *else*).

Gostou do conteúdo? Compartilha aí!

Dict Comprehensions com *if*

(<https://pythonacademy.com.br/blog/dict-comprehensions-no-python>)

Podemos adicionar lógica condicional à construção do dicionário resultante do nosso *Dict Comprehension*.

Podemos adicionar uma expressão condicional em três posições distintas:

Na construção da **chave**. Sintaxe:

```
1 {chave if condicao: valor for elemento in iteravel}
```

Na expressão que definirá o valor da chave:

```
1 {chave: valor if condicao for elemento in iteravel}
```

Ao final da expressão, filtrando os dados do iterável:

```
1 {chave: expressao for elemento in iteravel if condicao}
```

Podendo ser um **mix** da primeira versão, segunda ou terceira (*Aí o bagulho fica louco!*).

Por exemplo, se quisermos filtrar o dicionário de carros esportivos acima, pegando apenas aqueles com valor superior à R\$ 1.000.000,00?

Gostou do conteúdo? Compartilha aí!



```
1 import locale
2
3 # Configura o locale pra Português do Brasil (pt_BR)
4 locale.setlocale(locale.LC_MONETARY, 'pt_BR.utf8')
5
6 carros_esportivos = {
7     'ferrari': 1299000,
8     'lamborghini': 1100000,
9     'porsche': 759000
10 }
11
12 dict_saida = {
13     chave: f'{chave.upper()}: {locale.currency(valor)}'
14     for chave, valor in carros_esportivos.items() if valor > 1000000
15 }
```

Resultaria em:

```
{
    'ferrari': 'FERRARI: R$ 1299000,00',
    'lamborghini': 'LAMBORGHINI: R$ 1100000,00'
}
```

Resultado

Dict Comprehensions com vários if

E se quisermos alterar a chave e o valor na mesma expressão? 🤔

Podemos fazer da seguinte forma:

Gostou do conteúdo? Compartilha aí!



```

1  import locale
2
3  # Configura o locale pra Português do Brasil (pt_BR)
4  locale.setlocale(locale.LC_MONETARY, 'pt_BR.utf8')
5
6  carros_esportivos = {
7      'ferrari': 1299000,
8      'lamborghini': 1100000,
9      'porsche': 759000
10 }
11
12 dict_saida = {
13     chave if valor > 1000000 else f'{chave}-valor-abaixo':
14         f'{chave.upper()}: {locale.currency(valor)}'
15         if valor > 1000000 else f'{chave.upper()}: Valor abaixo de R$ 1.000.000,00'
16     for chave, valor in carros_esportivos.items()
17 }

```

Nesse exemplo:

- ◆ Chave será f'{chave}-valor-abaixo' caso valor seja menor que 1000000.
- ◆ Valor será f'{chave.upper()}: Valor abaixo de R\$ 1.000.000,00' caso valor seja menor que 1000000.

O resultado seria:

```

{
    'ferrari': 'FERRARI: R$ 1299000,00',
    'lamborghini': 'LAMBORGHINI: R$ 1100000,00',
    'porsche-valor-abaixo': 'Valor abaixo de R$ 1.000.000,00'
}




```

Gostou do conteúdo? Compartilha aí!

Resultado

Confundo, aqui vai uma dica.




<https://api.whatsapp.com/send?phone=5511999999999>

python@python.com

<https://t.me/python>

Percebeu o quão “embolado” ficou o código?

Não é por que o Python possibilite isso, que seja uma boa ideia utilizá-lo dessa forma.

Nunca se esqueçam do primeiro Zen do Python: “Bonito é melhor que feio”.

Ainda não conhece o explicativo do “Zen do Python” da Python Academy? 🤖

Conheça o Zen of Python clicando aqui (<https://pythonacademy.com.br/zen-of-python>) e se possível, o tatue no braço.

Brincadeira, só tatuá-lo já serve! 😄

Conclusão

Nesse *post* vimos como podemos usar *dict comprehensions* para criar e manipular dicts de uma maneira poderosa e eficiente.

Vimos quão eficiente essa técnica pode ser e as diversas formas de utilizá-la.

Agora que você está craque em *Dict Comprehensions*, ***que tal começar a utilizá-lo?***

Então... **Mão na massa!** 💪💪

Até o próximo *post*!

Gostou do conteúdo? Compartilha aí!





Autor

Por Vinícius Ramos em 14/01/2021

"Porque o Senhor dá a sabedoria, e da sua boca vem a inteligência e o entendimento. - Provérbios 2:6"

Minhas redes:

 (<https://www.linkedin.com/in/vinicius-aramos>) 

(<mailto:vinicius.ramos@pythonacademy.com.br>) 

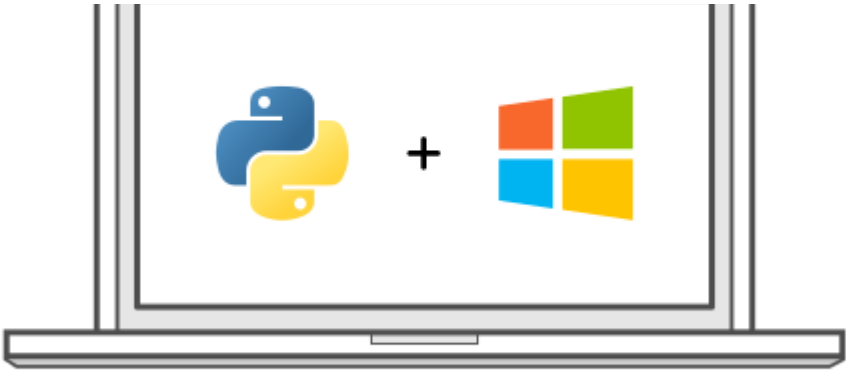
(<https://github.com/viniciusramos91>) 

(<https://stackoverflow.com/users/6775679/viniciusramos91>)

Continue aprendendo!

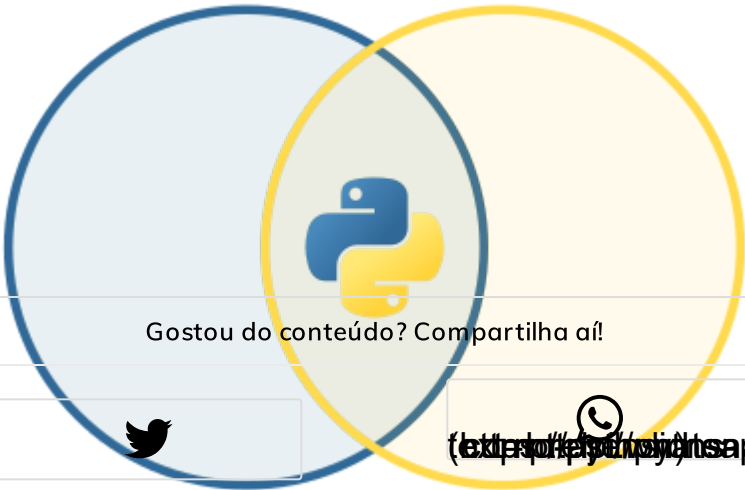
Gostou do conteúdo? Compartilha aí!





COMO INSTALAR O PYTHON NO WINDOWS

(/blog/como-instalar-python-no-windows)



SETS NO PYTHON

(/blog/sets-no-python)



Gostou do conteúdo? Compartilha aí!

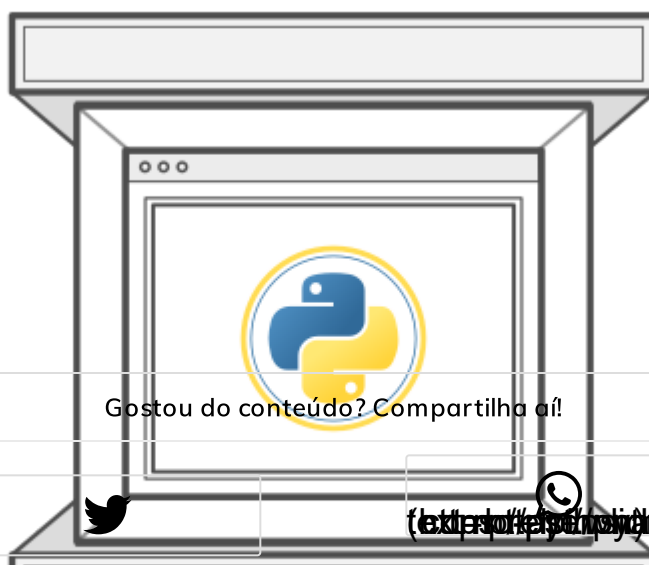


(<https://pythonacademy.com.br/blog/sets-no-python>)



LIST COMPREHENSIONS NO PYTHON

(/blog/list-comprehensions-no-python)



Gostou do conteúdo? Compartilha aí!



(<https://pythonacademy.com.br/blog/list-comprehensions-no-python>)






PYTHON E VIRTUALENV: COMO PROGRAMAR EM AMBIENTES VIRTUAIS (https://pythonacademy.com.br/blog/python-e-virtualenv-como-programar-em-ambientes-virtuais)

(/blog/python-e-virtualenv-como-programar-em-ambientes-virtuais)

Aprenda Python de Verdade!

Cadastre o seu **melhor email** para receber gratuitamente conteúdos exclusivos e muito mais!

 EU QUERO!

Gostou do conteúdo? Compartilha aí!



Mais

Início (/)

Zen of Python (/zen-of-python)

Blog (/blog/)

Sliders (/sliders/)

Sobre (/sobre)

 Loja Loja (/loja/)

Ebooks

Desenvolvimento Web com Python e Django (/ebooks/desenvolvimento-web-com-python-e-django/)

Cursos Parceiros

Curso Completo de Python (<https://go.hotmart.com/A55372730Q>)

Dominando Git e Github (<https://go.hotmart.com/V54192092I>)

HTML5, CSS3 e Bootstrap 4 (<https://go.hotmart.com/L54201537M>)

Gostou do conteúdo? Compartilha aí!



© 2021 Python Academy

*"Porque o Senhor dá a **sabedoria**, e da sua boca vem a **inteligência** e o **entendimento**" Pv 2:6*

Gostou do conteúdo? Compartilha aí!