

# Python

## Curso Intensivo

Prof. Cláudio Fleury  
Abr-22

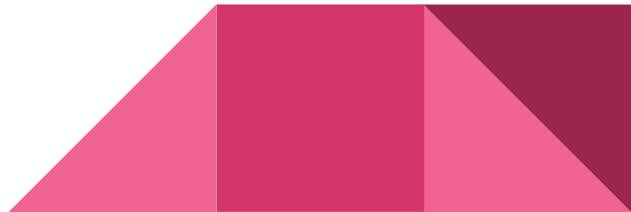
Curso baseado no  
livro de Eric Mattes,  
**Python Crash  
Course**

Cap.6 –  
Dicionários

# Conteúdo

1. Um Dicionário Simples
2. Acessando Dados do Dicionário
3. Removendo Pares **chave-valor**
4. Resumo

*Dicionários  
permitem a  
conexão de  
pedaços de  
informações  
relacionadas...*



# Um Dicionário Simples

Considere um jogo com alienígenas que podem ter diferentes valores de cor e ponto. Este dicionário simples armazena informações sobre um alienígena em particular:

File Edit Selection Find View Goto Tools Project Preferences Help

FOLD

platypus\_doc.py

dimensoes2.py

carros3.py

alienigina.py

```
1 alien_0 = {'cor': 'verde', 'pontos': 5}
2 print(alien_0['cor'])
3 print(alien_0['pontos'])
```

verde

5

[Finished in 187ms]

# Definição

- Um dicionário em Python é uma coleção de pares **chave-valor**. Cada chave está conectada a um valor e você pode usar uma chave para acessar o valor associado a essa chave. O valor da chave pode ser um número, uma string, uma lista ou até mesmo outro dicionário
- Cada chave é conectada ao seu valor por **dois pontos** e os pares **chave-valor** individuais são separados por vírgulas. Você pode armazenar quantos pares **chave-valor** desejar em um dicionário

```
alien_0 = {'cor': 'verde', 'pontos': 5}
```



A partir do Python 3.7, os dicionários mantêm a ordem em que foram definidos. Ao percorrer seus elementos, você verá os elementos na mesma ordem em que foram adicionados

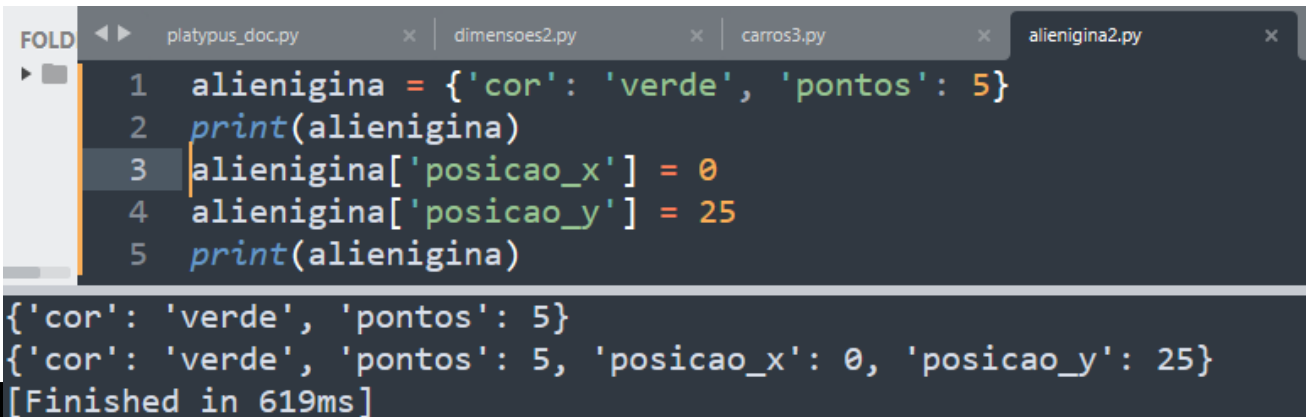
# Acessando Dados do Dicionário

- ◆ Forneça a chave como índice e o dicionário retorna o valor associado

```
alienigina = {'cor': 'verde', 'pontos': 5}
print(alienigina['cor'], alienigina['pontos'])
```

Verde 5

- ◆ Vamos colocar o alienígena na borda esquerda da tela, 25 pixels abaixo do topo. As coordenadas da tela começam no canto superior esquerdo da tela:  $x = 0$  e  $y = 25$



```
FOLD  platypus_doc.py x dimensoes2.py x carros3.py x alienigina2.py x
1 alienigina = {'cor': 'verde', 'pontos': 5}
2 print(alienigina)
3 alienigina['posicao_x'] = 0
4 alienigina['posicao_y'] = 25
5 print(alienigina)

{'cor': 'verde', 'pontos': 5}
{'cor': 'verde', 'pontos': 5, 'posicao_x': 0, 'posicao_y': 25}
[Finished in 619ms]
```

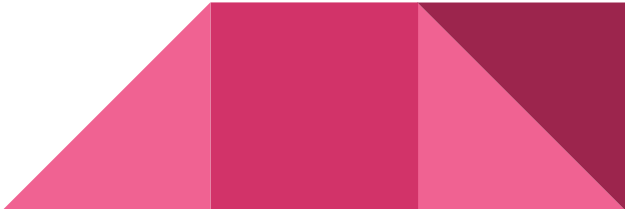
Dicionário vazio:  
`alien_0 = { }`

# Acessando Dados do Dicionário

- ◆ Usando **chave** entre colchetes para recuperar o **valor** associado de um dicionário pode causar um problema se a **chave** especificada não existir no dicionário, ou seja, você receberá um erro:

```
alien_0 = {'cor': 'verde', 'veloc': 'lento'}  
print(alien_0['pontos'])
```

```
Traceback (most recent call last):  
File "alienigina.py", line 2, in <module>  
print(alien_0['pontos'])  
KeyError: 'pontos'
```



# Acessando Dados do Dicionário

- ◆ Usando o método **get(chave, msg)**

O método `get()` requer uma chave como primeiro argumento, como segundo argumento opcional, o valor a ser retornado se a chave não existir no dicionário.

```
FOLD  linguagens.py x
1  linguagem_favorita = {
2      'Márcia': 'java',
3      'João': 'python',
4      'Rubens': 'c',
5      'Ana': 'fortran'
6  }
7  print(linguagem_favorita)
8  linguagem = linguagem_favorita['João'].title()
9  print(f"A linguagem favorita do João é {linguagem}.")
10 linguagem = linguagem_favorita.get('Maria', '**Usuário não cadastrado**')
11 print(f"A linguagem favorita do Maria é {linguagem}.")
```

```
{'Márcia': 'java', 'João': 'python', 'Rubens': 'c', 'Ana': 'fortran'}
A linguagem favorita do João é Python.
A linguagem favorita do Maria é **Usuário não cadastrado**.
[Finished in 188ms]
```

## Removendo Pares **chave-valor**

- ◆ Removendo o par da chave 'pontos'

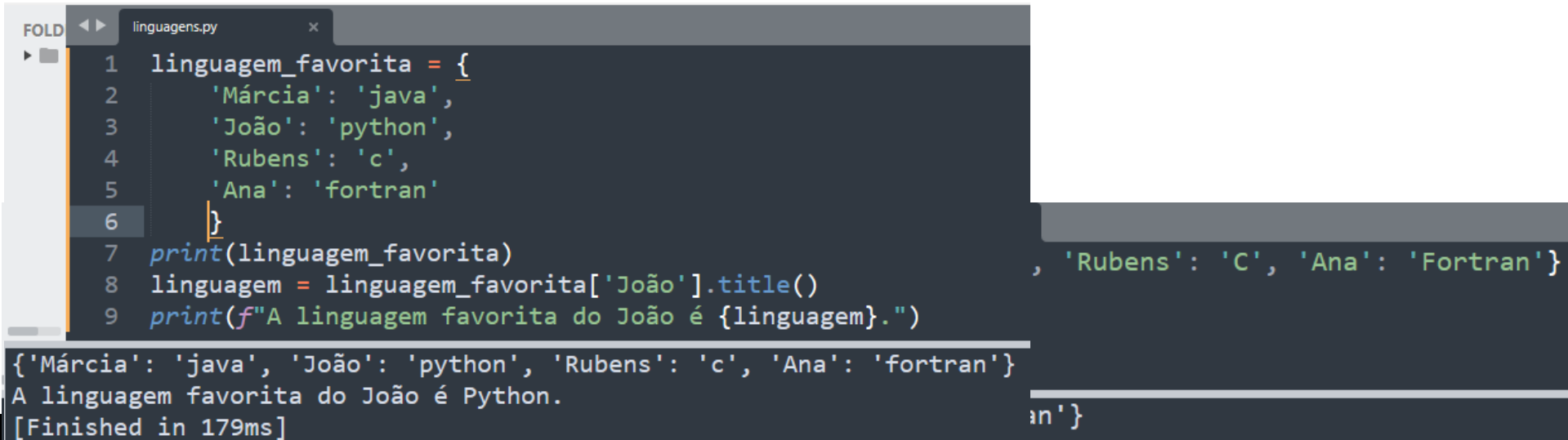
```
FOLD  ▶  platypus_doc.py  x  dimensoes2.py  x  carros3.py  x  alienigina3.py  x
1  alienigina = {'cor': 'verde', 'pontos': 5, 'posicao_x': 0,
2  print(alienigina)
3  del alienigina['pontos']
4  print(alienigina)
```

```
{'cor': 'verde', 'pontos': 5, 'posicao_x': 0, 'posicao_y': 25}
{'cor': 'verde', 'posicao_x': 0, 'posicao_y': 25}
[Finished in 174ms]
```



# Exemplo

- ◆ O exemplo anterior armazenou diferentes tipos de informações sobre um objeto, um alienígena de um jogo
- ◆ Podemos usar um dicionário para armazenar um tipo de informação sobre muitos objetos.



```
FOLD <> linguagens.py x
1 linguagem_favorita = {
2     'Márcia': 'java',
3     'João': 'python',
4     'Rubens': 'c',
5     'Ana': 'fortran'
6 }
7 print(linguagem_favorita)
8 linguagem = linguagem_favorita['João'].title()
9 print(f"A linguagem favorita do João é {linguagem}.")

{'Márcia': 'java', 'João': 'python', 'Rubens': 'c', 'Ana': 'fortran'}
A linguagem favorita do João é Python.
[Finished in 179ms]
```

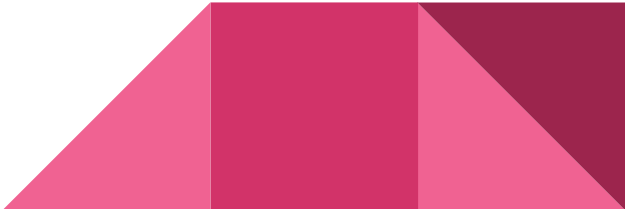
# Exercícios

**1. Testes condicionais:** escreva uma série de testes condicionais. Imprimir uma declaração descrevendo cada teste e sua previsão para os resultados de cada teste. Seu código deve ficar algo assim:

```
carro = 'subaru'
print('Carro é 'subaru'? Eu prevejo True.')
print(carro == 'subaru')
print('Carro é 'audi'? Eu prevejo False.')
print(carro == 'audi')
```

- Observe atentamente seus resultados e certifique-se de entender por que cada linha avalia como Verdadeiro ou Falso.
- Crie pelo menos dez testes. Tenha pelo menos cinco testes avaliados como Verdadeiro e outros cinco testes avaliam como Falso.

**2. Mais testes condicionais:** Crie pelo menos um resultado Verdadeiro e um Falso para cada um dos testes seguintes:

- Igualdade e desigualdade com strings
  - Usando o método lower()
  - Numéricos envolvendo igualdade e desigualdade, maior que menor que, maior ou igual a, e menor ou igual a
  - Usando a palavra-chave **and** e a palavra-chave **or**
  - Se um item está em uma lista
  - Se um item não está em uma lista
- 

# Exercícios

- 1. Pessoa:** Use um dicionário para armazenar informações sobre uma pessoa que você conhece. Armazene seu primeiro nome, sobrenome, idade e a cidade em que moram. Você deve ter chaves como `first_name`, `last_name`, `age` e `city`. Imprimir cada informação armazenada em seu dicionário.
- 2. Números favoritos:** use um dicionário para armazenar os números favoritos das pessoas. Pense em cinco nomes e use-os como chaves em seu dicionário. Pense em um número favorito para cada pessoa e armazene cada um como um valor em seu dicionário. Imprimir o nome de cada pessoa e seu número favorito. Para se divertir ainda mais, pesquise alguns amigos e obtenha alguns dados reais para o seu programa.
- 3. Glossário:** Um dicionário Python pode ser usado para modelar um dicionário real. No entanto, para evitar confusão, vamos chamá-lo de glossário.
  - Pense em cinco palavras de programação que você aprendeu nos capítulos anteriores. Use essas palavras como chaves em seu glossário armazene seus significados como valores.
  - Imprima cada palavra e seu significado como saída bem formatada. Você pode imprimir a palavra seguida de dois pontos e depois o seu significado, ou imprimir a palavra em uma linha e, em seguida, imprimir seu significado recuado em uma segunda linha. Use o caractere de nova linha (`\n`) para inserir uma linha em branco entre cada par palavra-significado em sua saída.

# Resumo

- Definição
- Dicionário Simples
- Constantes booleanas: True, False
- Sentença **if** simples, composta e cadeia
- Sentença **if** com listas

