16 - Função Lambda e List e Dict Compreehension - Exercícios-Resolvidos

July 25, 2022

1 Funções Lambda

[3]:
```
# Exercício 1 - Escreva uma função lambda que adiciona 5 a um número passado␣
↪como argumento para a função
add_5 = lambda x: x + 5
add_5(12)
```

[3]:
```
17
```

[5]:
```
# Exercício 2 - Escreva uma função utilizando "def" e dentro dela utilize␣
↪lambda.
# A função deve se chamar "dividir_por" e a partir dela deve ser possível criar␣
↪"funções filhas", exemplo: "dividir_por_2", "dividir_por_3"
# O número da função dividir_por vai ser o denominador.
def dividir_por(num):
    return lambda x: x / num
dividir_por_2 = dividir_por(2)
dividir_por_3 = dividir_por(3)
dividir_por_10 = dividir_por(10)
print(dividir_por_2(10))
print(dividir_por_3(18))
print(dividir_por_10(100))
```

```
5.0
6.0
10.0
```

[14]:
```
# Exercício 3 - Escreva uma função lambda que receba uma palavra como parâmetro␣
↪e retorne True se essa palavra começar com a letra "a" ou "A"
checar_palavra_comeca_com_a = lambda x: x[0].lower() == 'a'
print(checar_palavra_comeca_com_a("amor"))
print(checar_palavra_comeca_com_a("Amor"))
print(checar_palavra_comeca_com_a("Banana"))
```

```
True
True
```

```
False
```

[19]:
```
# Exercício 4 - Escreva uma função lambda para ordenar a lista abaixo em ordem alfabética
nomes = ['Jeniffer', 'Diego', 'Penélope', 'Thomas', 'Bruna', 'Emerson']
nomes.sort(key=lambda x: x[0])
nomes
```

[19]:
```
['Bruna', 'Diego', 'Emerson', 'Jeniffer', 'Penélope', 'Thomas']
```

[20]:
```
# Exercício 5 - Escreva uma função lambda para ordenar a lista do exercício anterior pela última letra do nome
nomes.sort(key=lambda x: x[-1])
nomes
```

[20]:
```
['Bruna', 'Penélope', 'Emerson', 'Diego', 'Jeniffer', 'Thomas']
```

[22]:
```
# Exercício 6 - Escreva uma função lambda para ordenar de forma descendente a lista abaixo pelo segundo item das tuplas
notas = [('Inglês', 8), ('Geografia', 5), ('Matemática', 10), ('Português', 7), ('História', 3)]
notas.sort(key=lambda x: x[1], reverse=True)
notas
```

[22]:
```
[('Matemática', 10),
 ('Inglês', 8),
 ('Português', 7),
 ('Geografia', 5),
 ('História', 3)]
```

2 List/Dict Comprehension

Utilize list/dict comprehension nos exercícios abaixo

[23]:
```
# Exercício 7 - Crie uma lista idêntica a lista abaixo
numeros = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
numeros2 = [x for x in numeros]
numeros2
```

[23]:
```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

[25]:
```
# Exercício 8 - Eleve ao quadrado todo número da lista "numeros" abaixo e retorne uma nova lista
numeros_ao_quadrado = [x ** 2 for x in numeros]
numeros_ao_quadrado
```

[25]:
```
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
```

[26]:
```
# Exercício 9 - O mesmo que o exercício 8, mas acrescente na nova lista, apenas␣
↪se o número elevado ao quadrado for maior que 30.
numeros_ao_quadrado_se_30 = [x ** 2 for x in numeros if x**2 > 30]
numeros_ao_quadrado_se_30
```

[26]:
```
[36, 49, 64, 81, 100]
```

[29]:
```
# Exercício 10 - Crie um dicionário a partir da lista abaixo, com a mesma chave␣
↪e valor (exemplo: "x":"x")
estados=['SP', 'RJ', 'BA']
dict_estados = {x:x for x in estados}
dict_estados
```

[29]:
```
{'SP': 'SP', 'RJ': 'RJ', 'BA': 'BA'}
```

[28]:
```
# Exercício 11 - Crie um dicionário com as chaves e valores abaixo.
keys = ['Inglês', 'Português', 'Matemática', 'Geografia']
values = [7, 4, 9, 8]
notas = {k:v for (k,v) in zip(keys, values)}
notas
```

[28]:
```
{'Inglês': 7, 'Português': 4, 'Matemática': 9, 'Geografia': 8}
```

[33]:
```
# Exercício 12 - Crie um dicionário a partir do range abaixo, onde cada número␣
↪do range é a chave e cada número dividido por 100 é o valor.
rng = range(100, 10000, 500)
dict1 = {x:x/100 for x in rng}
dict1
```

[33]:
```
{100: 1.0,
 600: 6.0,
 1100: 11.0,
 1600: 16.0,
 2100: 21.0,
 2600: 26.0,
 3100: 31.0,
 3600: 36.0,
 4100: 41.0,
 4600: 46.0,
 5100: 51.0,
 5600: 56.0,
 6100: 61.0,
 6600: 66.0,
 7100: 71.0,
 7600: 76.0,
 8100: 81.0,
 8600: 86.0,
```

```
9100: 91.0,
9600: 96.0}
```