

# ASIMOV

---

## map ()

map () é uma função que leva em dois argumentos: uma função e uma sequência iterable. Na forma: map(função, sequência) O primeiro argumento é o nome de uma função e a segunda uma sequência (por exemplo, uma lista). map() aplica a função a todos os elementos da sequência. Ele retorna uma nova lista com os elementos alterados por função.

Quando fomos sobre a compreensão da lista, criamos uma pequena expressão para converter Fahrenheit a Celsius. Vamos fazer o mesmo aqui, mas usando map. Começaremos com duas funções:

```
In [8]: def fahrenheit(T):  
        return ((float(9)/5)*T + 32)  
        def celsius(T):  
            return (float(5)/9)*(T-32)  
  
        temp = [0, 22.5, 40, 100]
```

Agora vamos ver o map() em ação:

```
In [12]: F_temps = list(map(fahrenheit, temp))  
  
        # Mostra  
        F_temps
```

```
Out[12]: [32.0, 72.5, 104.0, 212.0]
```

```
In [13]: # Converte devolta  
        list(map(celsius, F_temps))
```

```
Out[13]: [0.0, 22.5, 40.0, 100.0]
```

No exemplo acima, não usamos uma expressão lambda. Ao usar lambda, não teríamos que definir e nomear as funções fahrenheit() e celsius().

```
In [14]: list(map(lambda x: (5.0/9)*(x - 32), F_temps))
```

```
Out[14]: [0.0, 22.5, 40.0, 100.0]
```

Ótimo! Nós obtivemos o mesmo resultado! O uso do map() é muito mais comumente usado com expressões lambda, já que todo o propósito do map() é economizar esforço ao ter que criar manual para loops.

map() pode ser aplicado a mais de um iterable. Os iteráveis devem ter o mesmo comprimento.

Por exemplo, se estamos trabalhando com duas listas-map() aplicará sua função lambda aos elementos das listas de argumentos, ou seja, aplica-se primeiro aos elementos com o índice 0, e depois aos elementos com o 1º índice até o que o índice N seja alcançado.

Por exemplo, mapeamos uma expressão lambda para duas listas:

In [16]:

```
a = [1,2,3,4]
b = [5,6,7,8]
c = [9,10,11,12]

list(map(lambda x,y:x+y,a,b))
```

Out[16]: [6, 8, 10, 12]

In [18]:

```
list(map(lambda x,y,z:x+y+z, a,b,c))
```

Out[18]: [15, 18, 21, 24]

Podemos ver no exemplo acima que o parâmetro x obtém seus valores da lista a, enquanto y obtém seus valores de b e z da lista c. Vá em frente e teste com seu próprio exemplo para se certificar de que compreende o mapeamento para mais do que um iterable.

Bom trabalho! Você deve agora ter uma compreensão básica da função map ().