map()

map() یک تابع داخلی در پایتون است که دو یا بیشتر آرگومان دریافت می‌کند: یک تابع و یک یا چند آیتم قابل تکرار در شکل زیر:

```
map(function, iterable, ...)
```

map() یک *ایتریتور* برمی‌گرداند - به عبارت دیگر، map() یک شیء ویژه‌ای را برمی‌گرداند که به تدریج یک نتیجه را در هر زمان به صورت لازم تولید می‌کند. ما در فصول آینده بیشتر درباره ایتریتورها و ژنراتورها صحبت خواهیم کرد. در حال حاضر، به دلیل اینکه مثال‌های ما خیلی کوچک هستند، ما map() را به صورت یک لیست تبدیل می‌کنیم تا نتایج را بلافاصله ببینیم.

وقتی ما درباره ترکیبات لیست صحبت می‌کردیم، یک List Comprehension برای تبدیل درجه سلسیوس به فارنهایت ایجاد کردیم. همین کار را اینجا انجام می‌دهیم، اما از map استفاده می‌کنیم:

In [2]:

```
def fahrenheit(celsius):
    return (9/5) * celsius + 32

temps = [0, 22.5, 40, 100]
```

حالا بیایید map() را در عمل ببینیم:

In [5]:

```
F_temps = map(fahrenheit, temps)

# [*F_temps]

list(F_temps)
```

Out[5]:

```
[32.0, 72.5, 104.0, 212.0]
```

در مثال بالا، map() تابع فارنهایت را به هر آیتم در temps اعمال می‌کند. با این حال، ما نیازی به تعریف توابع قبلی نداریم؛ بلکه می‌توانیم از یک عبارت لامبدا استفاده کنیم.

In [6]:

```
list( map(lambda x: (9/5) * x + 32, temps) )
```

Out[6]:

```
[32.0, 72.5, 104.0, 212.0]
```

عالی! ما به همان نتیجه رسیدیم! استفاده از map() با عبارات لامبدا بسیار رایج‌تر است، زیرا هدف کلی map() از صرفه‌جویی در زمان و تلاش برای ایجاد حلقه‌های دستی است.

map() با چندین تکرارشوند

map() می‌تواند بیش از یک آیتریبل دریافت کند. تکرارشوندها باید همان طول باشند - در صورتی که طول آن‌ها برابر نباشد، map() همچنین خواهد ماند تا زمانی که کوتاه‌ترین تکرارشوند تمام شود.

به عنوان مثال، اگر تابع ما سعی در اضافه کردن دو مقدار **x** و **y** داشته باشد، می‌توانیم یک لیست از مقادیر **x** و یک لیست دیگر از مقادیر **y** را به map() ارسال کنیم. تابع (یا لامبدا) هر بار مقدار 0-ام را از هر لیست دریافت می‌کند، سپس مقدار 1-ام و به همین ترتیب تا رسیدن به مقدار n-ام.

In [10]:

```
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[10]:

```
[6, 8, 10, 12]
```

In [11]:

```
# Now all three lists
list( map(lambda x, y, z: x + y + z , a, b, c))
```

Out[11]:

```
[15, 18, 21, 24]
```

در مثال بالا می‌بینیم که پارامتر **x** مقادیر خود را از لیست **a** دریافت می‌کند، در حالی که **y** مقادیر خود را از **b** و **z** از لیست **c** دریافت می‌کند. حالا شما می‌توانید با مثال خودتان بازی کنید تا اطمینان حاصل کنید که به درستی نحوه تطبیق با بیش از یک تکرارشوند را درک کرده‌اید.

کار عالی! حالا باید یک درک اولیه از تابع map() داشته باشید.