

Loops

5. حلقه ها

For Loop (حلقه For)

از حلقه for برای پیمایش در یک بازه یا مجموعه استفاده کنید.

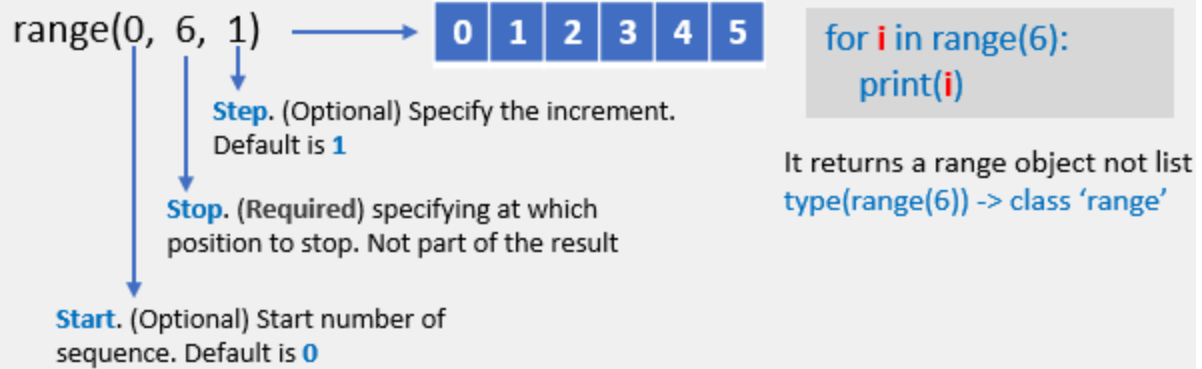
برای اجرای مکرر مجموعه‌ای از کدها به تعداد مشخص، می‌توان از تابع `range()` استفاده کرد. این تابع یک دنباله از اعداد ایجاد می‌کند که به صورت پیش فرض:

- از عدد 0 شروع می‌شود،
- با گام 1 افزایش می‌یابد،
- و پیش از رسیدن به عدد انتهایی مشخص شده متوقف می‌شود.

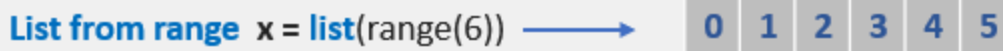
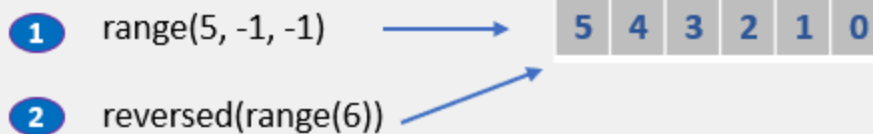
سینتکس این تابع به صورت زیر است: `range(start, stop, step_size)`

How to use Python range(start, stop, step)

range() returns the immutable sequence of numbers starting from the given **start integer** to the **stop-step**. Each number is incremented by adding step value to its preceding number



Reverse range/ Decrementing



range(-1, -11, -1)	Negative range from -1 to -10
range(start, stop+step, step)	Generate an inclusive range
range(0, 10)[5]	Access 5th number of a range()
range(10)[3:8]	Slice a range to from index 3 to 8
range(5).start, range(5).stop range(5).step	Access range() attributes

PYnative.com

```

1 | # Looping from 0 to 4
2 | for i in range(5):
3 |     print("Number:", i) # Output: 0, 1, 2, 3, 4

```

زمانی که از تابع `range()` استفاده می‌کنیم، چه اتفاقی می‌افتد؟

تابع `range()` در واقع یک دنباله (Sequence) از اعداد تولید می‌کند. این دنباله به صورت دینامیک (در زمان اجرا) ایجاد می‌شود و به جای ذخیره‌شدن به عنوان یک لیست واقعی در حافظه، تنها بازه و گام‌های تولید اعداد را نگهداری می‌کند.

در صورتی که این دنباله را به لیست تبدیل کنیم، اعداد به صورت یک لیست واقعی در حافظه ذخیره می‌شوند. برای مثال:

```
1 | numbers = list(range(5))
2 | print(numbers)
3 | # output: [0, 1, 2, 3, 4]
```

در این صورت، خروجی `[0, 1, 2, 3, 4]` خواهد بود که یک لیست واقعی است.

بدون استفاده از `list()`، تابع `range()` تنها یک دنباله از اعداد را تولید می‌کند که به صورت مستقیم می‌توان از آن در حلقه‌ها یا دیگر عملیات‌ها استفاده کرد. مثلاً:

```
1 | for i in range(5):
2 |     print(i)
```

تحلیل عملکرد:

۱. با اجرای `range(5)`، دنباله‌ای از اعداد از ۰ تا ۴ (یعنی $n-1$) به طور دینامیک تولید می‌شود.

۲. در هر گام از حلقه، مقدار `i` یکی از مقادیر این دنباله را دریافت می‌کند:

◦ ابتدا $i = 0$ ، سپس $i = 1$ ، و به همین ترتیب تا $i = 4$.

۳. زمانی که همه مقادیر پیمایش شدند، حلقه به پایان می‌رسد.

تابع `len()`

تابع `len()` تعداد عناصر موجود در یک شیء را بازمی‌گرداند. این تابع برای انواع مختلفی از اشیاء در پایتون قابل استفاده است، مانند:

- رشته‌ها (string): تعداد کاراکترهای موجود در رشته را برمی‌گرداند.
- لیست‌ها (list): تعداد عناصر موجود در لیست را بازمی‌گرداند.
- سایر انواع مجموعه‌ها (collections): تعداد اعضای آن‌ها را ارائه می‌دهد.

مثال: محاسبه تعداد کاراکترهای یک رشته

اگر شیء ورودی یک رشته باشد، تابع `len()` تعداد کاراکترهای موجود در آن را بازمی‌گرداند. به مثال زیر توجه کنید:

```
1 myString = "Hello"
2 print(len(myString)) # output: 5
```

حلقه While

از حلقه `while` برای اجرای کد تا زمانی که یک شرط برقرار باشد استفاده کنید.

```
1 # Countdown loop
2 count = 3
3 while count > 0:
4     print("Counting down:", count)
5     count -= 1 # Decrease the count by 1
```