for döngüsü: **Belirli sayıda işlemlerin tekrarlanması için kullanılan döngülerdir. for döngüleri başlangıç ve bitiş değerleri arasında artış miktarına göre istenilen sayıda tekrar yapar.**

Ör: 1'den 5' kadar olan sayıları yazdıralım.

```
print(1)
print(2)
print(3)
print(4)
print(5)
```

Çıktı:
1
2
3
4
5

→Ancak, 1'den 100'e kadar yazmak gerekirse böyle bir çözüm yolu doğru olmayacaktır! Bu durumda döngü yapıları tercih edilmelidir. Python dilinde döngü için while ve for yapıları kullanılır.

Ör: 1'den 100'e kadar olan sayıları yazdıralım. Değişkenimiz n olsun:

```
for n in range(1,100):
     print(n)
```

1
2
.
.
98
99

Ör: Şimdi de 1'den 100'e kadar olan tek sayıları yazdıralım. Değişkenimiz yine n olsun:

```
for n in range(1,100,2):
     print(n)
```

Çıktı:
1
3
5
.
.
.
97
99

→Açıklama: range(1,100,2) ifadesindeki 1 başlangıç sayısıdır. Eğer burası boş bırakılırsa sayı otomatik olarak sıfırdan başlar. 100 ise yazılacak sayıların sınırıdır. 100 çıktıya dâhil değildir. 2 ise artış miktarını gösterir. Yani sayıyı 2'şer arttırır.

Örnekler:

range(10) → **0,1,2,3,4,5,6,7,8,9** **→başlangıç ve artış değeri yok. Sadece bitiş değeri var.**

range(1, 10) → **1,2,3,4,5,6,7,8,9** **→ artış değeri yok. Sadece başlangıç ve bitiş değeri var.**

range(1, 10, 2) → **1,3,5,7,9** **→ başlangıç, bitiş ve artış değeri var.**

range(10, 0, -1) → **10,9,8,7,6,5,4,3,2,1** **→ buradaki artış değeri eksiye doğru gitmektedir.**

range(10, 0, -2) → **10,8,6,4,2**

range(2, 11, 2) → **2,4,6,8,10**

range(-5, 5) → **−5,−4,−3,−2,−1,0,1,2,3,4**

range(1, 2) → **1**

range(1, 1) → **()**

range(1, -1) → **()**

range(1, -1, -1) → **1,0**

range(0) → **()**

Ör: 21'den 0'a kadar olan sayıları 3'er 3'er azaltarak yazdıralım.

```
for n in range(21,0,-3):
     print(n)
```

21
18
15
12
9
6
3

→ Yukarıdaki çıktıyı bir de yan yana gelecek şekilde yazdıralım.

```
for n in range(21,0,-3):
     print(n, end=" ")
```

21 18 15 12 9 6 3

Ör: 1'den 100' kadar olan sayıların toplamı:

```
top= 0
for i in range(1,100):
     top+= i
print(top)
```

4950

Ör: 10 ve 10'un üstleri yazdıran program:

```
for i in range(7):

  print("{}".format(10**i))
```

1
10
100
1000
10000
100000
1000000

Ör: Bir string değişkeni oluşturarak, string'teki her bir karakteri ayrı ayrı işleme ve yazdırma:

```
a = "kodlama"
for harf in a:
     print(harf, end=" ")
```

k o d l a m a

Ör: Bir string değişkeni oluşturarak, string'teki her bir karakteri ayrı ayrı işleme ve yazdırma:

```
sayılar = "12345"
for sayı in sayılar:
     print(int(sayı) * 2, end=" ")   #her karakteri tamsayıya çevirip 2 ile çarp
```

2 4 6 8 10

Ör:

```
sayılar = "1234567"

for i in sayılar:
    if int(i) > 3:
        print(i, end=" ")
```

4 5 6 7

→**Açıklaması:** Burada **sayılar** değişkeni oluşturduk. **"1234567"** ifadesindeki her bir karakteri ise ayırdık. Yani bu ifadenin içindeki tüm karakterler artık bağımsızlığını ilan etmiş durumda. Ayrıca her bir karakteri de **i** değişkene atadık. Burada i değişkenine atama işlemini **for i** yazarak, her bir karakteri ayırma işlemini ise **in sayılar** yazarak hallettik. İkisini birleştirdiğimizde ortaya **for i in sayılar** kodu çıkıyor. Artık 1 bağımsız 2 bağımsız 3 bağımsız 4 bağımsız 5 bağımsız 6 bağımsız 7 bağımsız bir karakter oldular. Ancak biz bu karakterleri **int(i)** kodunu yazarak tamsayıya çevirdik. Öyle ya matematiksel işlemler yapmak için verileri sayıya çevirmemiz gerekiyor. **if** komutuyla da 3'ten büyük olanları yazdırdık