


Python'da **list comprehension**, bir liste oluşturmanın kısa ve okunabilir bir yoludur. Geleneksel `for` döngüsü kullanarak bir liste oluşturmak yerine, **list comprehension** kullanarak aynı işlemi tek bir satırda gerçekleştirebilirsiniz.

Klasik Döngü

```
python


squares = []
for x in numbers:
    squares.append(x**2)
```

 Kodu kopyala

List Comprehension

```
python


squares = [x**2 for x in numbers]
```

 Kodu kopyala

Temel Yapı

```
python

[expression for item in iterable if condition]
```

 Kodu kopyala


- **expression:** Listeye eklenecek değer.
- **item:** Döngüdeki her bir eleman.
- **iterable:** Üzerinde döngü yapılan veri (liste, dizi, vb.).
- **condition** (*isteğe bağlı*): Elemanların listeye eklenip eklenmeyeceğini belirler.

Örnekler

1. Basit Bir Liste Oluşturma


```
python

numbers = [1, 2, 3, 4, 5]
squares = [x**2 for x in numbers]
print(squares)
# Çıktı: [1, 4, 9, 16, 25]
```

 Kodu kopyala

2. Koşul Kullanımı


python

 Kodu kopyala

```
numbers = [1, 2, 3, 4, 5, 6]
even_numbers = [x for x in numbers if x % 2 == 0]
print(even_numbers)
# Çıktı: [2, 4, 6]
```

3. Birden Fazla Döngü

python

 Kodu kopyala

```
pairs = [(x, y) for x in range(1, 3) for y in range(3, 5)]
print(pairs)
# Çıktı: [(1, 3), (1, 4), (2, 3), (2, 4)]
```

4. Koşul ve İfade Kullanımı

python

 Kodu kopyala

```
numbers = [1, 2, 3, 4, 5]
result = [x**2 if x % 2 == 0 else x**3 for x in numbers]
print(result)
# Çıktı: [1, 4, 27, 16, 125]
```