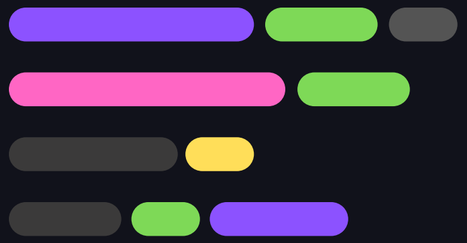
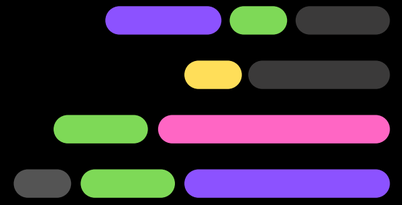
{TEMEL PROGRAMLAMA II}

Samsun Üniversitesi
Teknik Bilimler Meslek Yüksekokulu
Arka-Yüz Yazılım Geliştirme Pr.



Öğr. Gör. Furkan DURMUŞ

Bu Haftanın Ders Kazanımları



01 Lambda Fonksiyonlar

02 Özyinelemeli Fonksiyonlar

03 Uygulama Saati

04 Proje Sunum Saati

05 Proje Çalışma Saati

Lambda Fonksiyonlar

01

Lambda Fonksiyonlar

Lambda, Python'da tek satırlık fonksiyon yazmanın kısa yoludur. Python programlama dilinde ihtiyaç hâlinde def ifadesi kullanmadan isimsiz, tek satırlık, küçük fonksiyonlar da tanımlanabilir. Bu fonksiyonlara **anonim fonksiyonlar** veya tanımlarken kullanılan lambda ifadesi nedeniyle **lambda fonksiyonlar** denir.

01

Lambda Fonksiyonlar

Lambda, Python’da tek satırlık fonksiyon yazmanın kısa yoludur.

```python
def kare(x):
    return x * x
```

```python
kare = lambda x: x * x

print(kare(5))  # 25
```

01

Lambda Fonksiyonlar

```python
kare = lambda x: x * x

print(kare(5))  # 25
```

`lambda` **`parametreler`** `:` **`ifade`**

01

Lambda Fonksiyonlar

```python
def kare(x):
    return x * x

print(kare(5))
```

```python
kare = lambda x: x * x

print(kare(5))  # 25
```

01

Lambda Fonksiyonlar

Lambda ile **Matematiksel** işlem yapılabilir.

```
islem = lambda a, b: (a + b) / 2

print(islem(10, 20))
```

01

Lambda Fonksiyonlar

Lambda ile **Matematiksel** işlem yapılabilir.

Sayının küpünü döndüren lambda fonksiyonunu yazınız.

```
kup_hesapla = 
print(kup_hesapla(2))
```

01

Lambda Fonksiyonlar

Lambda ile **Matematiksel** işlem yapılabilir.

Sayının küpünü döndüren lambda fonksiyonunu yazınız.

```
kup_hesapla = lambda x: x ** 3
print(kup_hesapla(2))
```

01

Lambda Fonksiyonlar

Lambda ile **Matematiksel** işlem yapılabilir.

2 sayıyı çarpan lambda fonksiyonunu yazınız.

01

Lambda Fonksiyonlar

Lambda ile **Matematiksel** işlem yapılabilir.

2 sayıyı çarpan lambda fonksiyonunu yazınız.

```
carpma_islemi = lambda x, y: x * y
print(carpma_islemi(4, 6))
```

01

Lambda Fonksiyonlar

Lambda ile **Matematiksel** işlem yapılabilir.

Sayının 5 fazlasını döndüren lambda fonksiyonunu yazınız.

01

Lambda Fonksiyonlar

Lambda ile **Matematiksel** işlem yapılabilir.

Sayının 5 fazlasını döndüren lambda fonksiyonunu yazınız.

```
islem = lambda x: x + 5
print(islem(3))
```

01

Lambda Fonksiyonlar

Lambda ile **Karşılaştırma** yapılabilir.

```
pozitif_mi = lambda x: x > 0
print(pozitif_mi(-3))
```

01

Lambda Fonksiyonlar

Lambda ile **Karşılaştırma** yapılabilir.

İki sayının eşit olup olmadığını bool ile dönen lambda fonksiyonunu yazınız.

```
esit_mi = [redacted]
print(esit_mi(5, 5))
```

01

Lambda Fonksiyonlar

Lambda ile **Karşılaştırma** yapılabilir.

İki sayının eşit olup olmadığını bool ile dönen lambda fonksiyonunu yazınız.

```
esit_mi = lambda a, b: a == b
print(esit_mi(5, 5))
```

01

Lambda Fonksiyonlar

Lambda ile **Karşılaştırma** yapılabilir.

Sayının çift olup olmadığını bool ile dönen lambda fonksiyonunu yazınız.

01

Lambda Fonksiyonlar

Lambda ile **Karşılaştırma** yapılabilir.

Sayının çift olup olmadığını bool ile dönen lambda fonksiyonunu yazınız.

```
cift_mi = lambda x: x % 2 == 0
print(cift_mi(8))
```

01

Lambda Fonksiyonlar

Lambda ile **Karşılaştırma** yapılabilir.

Girilen sayının 0 ve 13 arasında olup olmadığını bool ile dönen lambda fonksiyonunu yazınız.

01

Lambda Fonksiyonlar

Lambda ile **Karşılaştırma** yapılabilir.

Girilen sayının 0 ve 13 arasında olup olmadığını bool ile dönen lambda fonksiyonunu yazınız.

```
aralikta_mi = lambda x: 1 <= x <= 12
print(aralikta_mi(15))
```

01

Lambda Fonksiyonlar

Lambda ile **Koşullu İfade(Ternary)** yapılabilir.

```
isaret_belirle = lambda x: "Pozitif" if x > 0 else "Negatif"
print(isaret_belirle(-5))
```

01

Lambda Fonksiyonlar

Lambda ile **Koşullu İfade(Ternary)** yapılabilir.

Sınav notu 50 ve üstü ise ‘Geçti’ değilse ‘Kaldı’ mesajını dönen lambda fonksiyonunu yazınız.

```
durum_belirle =
print(durum_belirle(70))
```

01

Lambda Fonksiyonlar

Lambda ile **Koşullu İfade(Ternary)** yapılabilir.

Sınav notu 50 ve üstü ise ‘Geçti’ değilse ‘Kaldı’ mesajını dönen lambda fonksiyonunu yazınız.

```
durum_belirle = lambda notu: "Geçti" if notu >= 50 else "Kaldı"
print(durum_belirle(70))
```

01

Lambda Fonksiyonlar

Lambda ile **Koşullu İfade(Ternary)** yapılabilir.

Girilen sayının tek mi çift mi olduğunu dönen lambda fonksiyonunu yazınız.

01

Lambda Fonksiyonlar

Lambda ile **Koşullu İfade(Ternary)** yapılabilir.

Girilen sayının tek mi çift mi olduğunu dönen lambda fonksiyonunu yazınız.

```
tek_cift_belirle = lambda x: "Çift" if x % 2 == 0 else "Tek"
print(tek_cift_belirle(9))
```

01

Lambda Fonksiyonlar

Lambda ile **Koşullu İfade(Ternary)** yapılabilir.

Girilen yaşa göre reşit mi değil mi olduğunu dönen lambda fonksiyonunu yazınız.

01

Lambda Fonksiyonlar

Lambda ile **Koşullu İfade(Ternary)** yapılabilir.

Girilen yaşa göre reşit mi değil mi olduğunu dönen lambda fonksiyonunu yazınız.

```
resit_mi = lambda yas: "Reşit" if yas >= 18 else "Reşit değil"
print(resit_mi(16))
```

01

Lambda Fonksiyonlar

Lambda ile **Fonksiyon** çağrılabilir.

```
mutlak_deger_al = lambda x: abs(x)
print(mutlak_deger_al(-20))
```

01

Lambda Fonksiyonlar

Lambda ile **Fonksiyon** çağrılabilir.

Girilen metnin uzunluğunu dönen lambda fonksiyonunu yazınız.

01

Lambda Fonksiyonlar

Lambda ile **Fonksiyon** çağrılabilir.

Girilen metnin uzunluğunu dönen lambda fonksiyonunu yazınız.

```
uzunluk_hesapla = lambda metin: len(metin)
print(uzunluk_hesapla("Python"))
```

01

Lambda Fonksiyonlar

Lambda ile **Fonksiyon** çağrılabilir.

Girilen sayıyı yuvarlayıp dönen lambda fonksiyonunu yazınız.

01

Lambda Fonksiyonlar

Lambda ile **Fonksiyon** çağrılabilir.

Girilen sayıyı yuvarlayıp dönen lambda fonksiyonunu yazınız.

```
yuvarla = lambda sayi: round(sayi)
print(yuvarla(3.7))
```

01

Lambda Fonksiyonlar

Lambda ile **Fonksiyon** çağrılabilir.

Gönderilen listedeki maksimum değerli sayıyı bulup dönen lambda fonksiyonunu yazınız.

01

Lambda Fonksiyonlar

Lambda ile **Fonksiyon** çağrılabilir.

Gönderilen listedeki maksimum değerli sayıyı bulup dönen lambda fonksiyonunu yazınız.

```
en_buyuk_degeri_bul = lambda liste: max(liste)
print(en_buyuk_degeri_bul([3, 8, 1]))
```

01

Lambda Fonksiyonlar

Lambda ile **list / tuple / dict** oluşturulabilir.

```
liste_olustur = lambda x: [x, x+1, x+2]
print(liste_olustur(5))
```

01

Lambda Fonksiyonlar

Lambda ile **list / tuple / dict** oluşturulabilir.

Gönderilen 2 sayıyı tuple yapıp dönen lambda fonksiyonunu yazınız.

01

Lambda Fonksiyonlar

Lambda ile **list / tuple / dict** oluşturulabilir.

Gönderilen 2 sayıyı tuple yapıp dönen lambda fonksiyonunu yazınız.

```
sonuc_tuple_olustur = lambda a, b: (a+b, a*b)
print(sonuc_tuple_olustur(3, 4))
```

01

Lambda Fonksiyonlar

Lambda ile **list / tuple / dict** oluşturulabilir.

Gönderilen isim ve soy ismi sözlük yapıp dönen lambda fonksiyonunu yazınız.

```
ogrenci_dict_olustur = lambda ad,soyad: {"isim": ad,"soyisim": soyad}
print(ogrenci_dict_olustur("Ali","Şahin"))
```

01

Lambda Fonksiyonlar

Lambda ile **slicing(parçalama)** yapılabilir.

```
ilk_uc_harf = lambda metin: metin[:3]
print(ilk_uc_harf("Python"))
```

01

Lambda Fonksiyonlar

Lambda ile **slicing(parçalama)** yapılabilir.

```
son_iki_harf = lambda metin: metin[-2:]
print(son_iki_harf("Python"))
```

01

Lambda Fonksiyonlar

Lambda ile **Mantıksal ifadeler** yapılabilir.

```
aralik_kontrol = lambda x: x > 0 and x < 100
print(aralik_kontrol(50))
```

01

Lambda Fonksiyonlar

Lambda ile **Mantıksal ifadeler** yapılabilir.

```
ozel_sayi_mi = lambda x: x == 0 or x == 1
print(ozel_sayi_mi(1))
```

01

Lambda Fonksiyonlar

Lambda ile **döngü** **yapılamaz**.

```
islem = lambda x: for i in range(x): print(i)
```

01

Lambda Fonksiyonlar

Lambda ile **döngü** **yapılamaz**.

```
islem = lambda x:    Expected expression
    for i in range(5):    Unexpected indentation
        print(i)
```

01

Lambda Fonksiyonlar

Lambda ile **döngü** **yapılamaz**.

```
islem = lambda x: while x > 0: x -= 1
```

01

Lambda Fonksiyonlar

Lambda ile **return ifadesi** aynı anda **kullanılmaz**.

```
islem = lambda x: return x*2
```

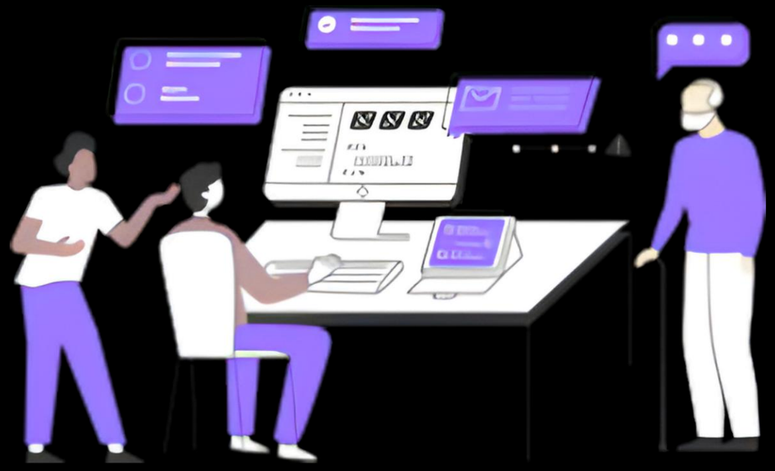
Özyinelemeli Fonksiyonlar

02

Özyinelemeli Fonksiyonlar

Bir fonksiyon, çözmek için tasarlandığı problemi çözerken kendi içinde yeniden kendini çağırabilir.

Fonksiyonun kendi kendini çağırmasına ***özyineleme*** *(recursion), bu tip fonksiyonlara ise* ***özyinelemeli*** *(recursive) fonksiyon denir.*

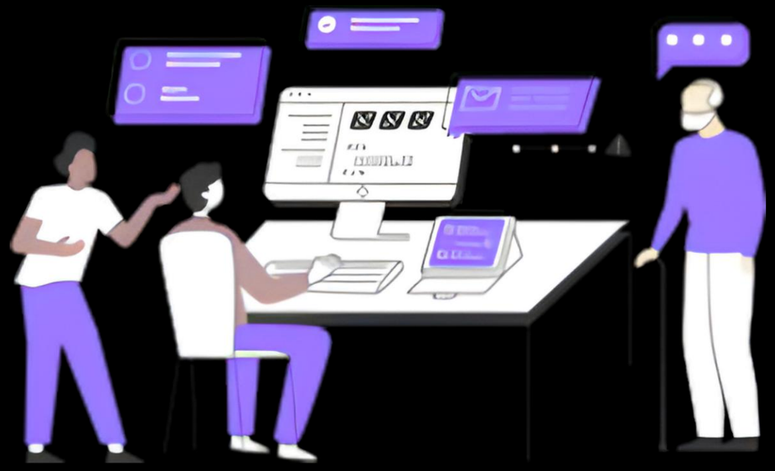


02

Özyinelemeli Fonksiyonlar

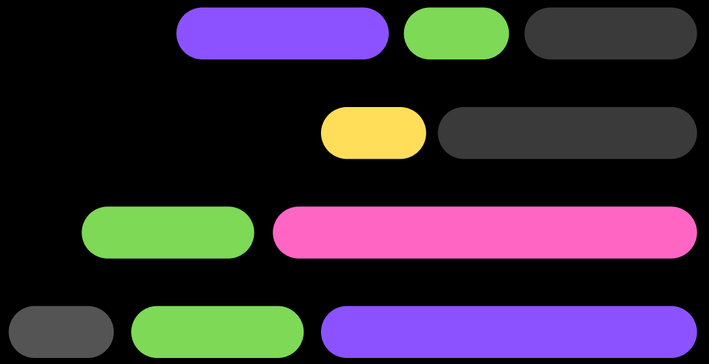
*Bir fonksiyonun kendi kendini çağırmasına **özyineleme** denir.*

```
def fonksiyon():
    fonksiyon()
```



02

Özyinelemeli Fonksiyonlar



Özyinelemeli fonksiyonlar, bazı durumlarda çok kullanışlı ve bazı problemlerin çözümünde elzem olsalar da fonksiyonlarda karmaşıklığı ve programlarda bellek kullanımını arttırır.

*Bu nedenle Python programlama dili özyineleme derinliğini varsayılan olarak **1000** ile sınırlandırır. Bu sınır aşıldığında fonksiyonunuz **RecursionError** (özyineleme hatası) verir.*

```
def fonksiyon():
    print('Merhaba')
    fonksiyon()
```

02

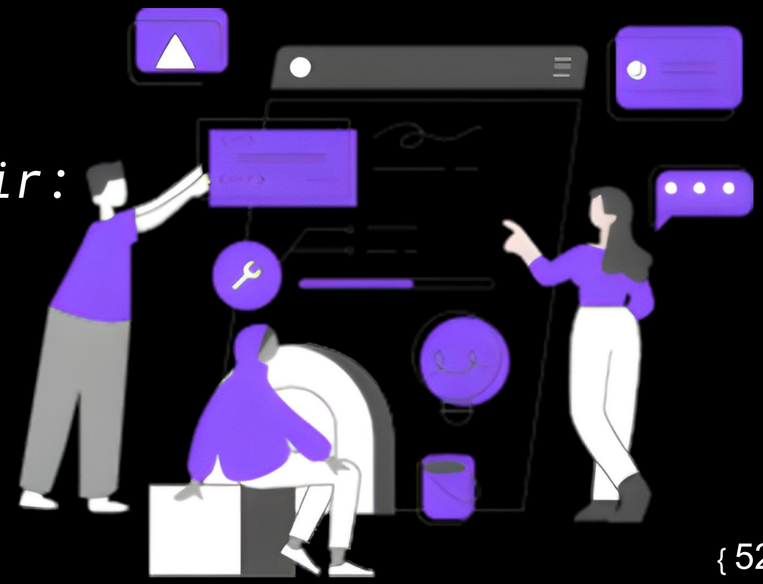
Özyinelemeli Fonksiyonlar

```python
def fonksiyon():
    print('Merhaba')
    fonksiyon()
```

Recursion'ın doğru çalışması için iki temel unsur gerekir:

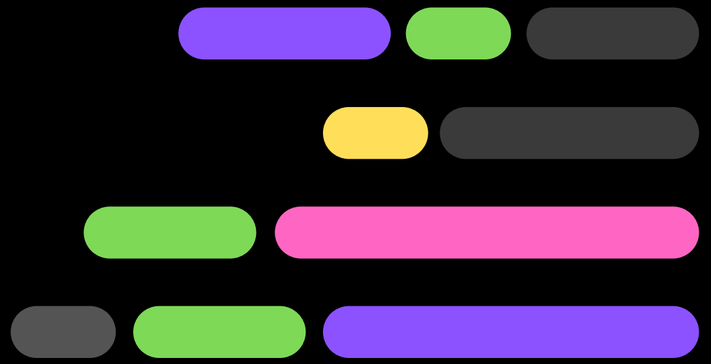
1- Base Case (Temel Durum)

2- Recursive Case (Özyinelemeli Durum)



02

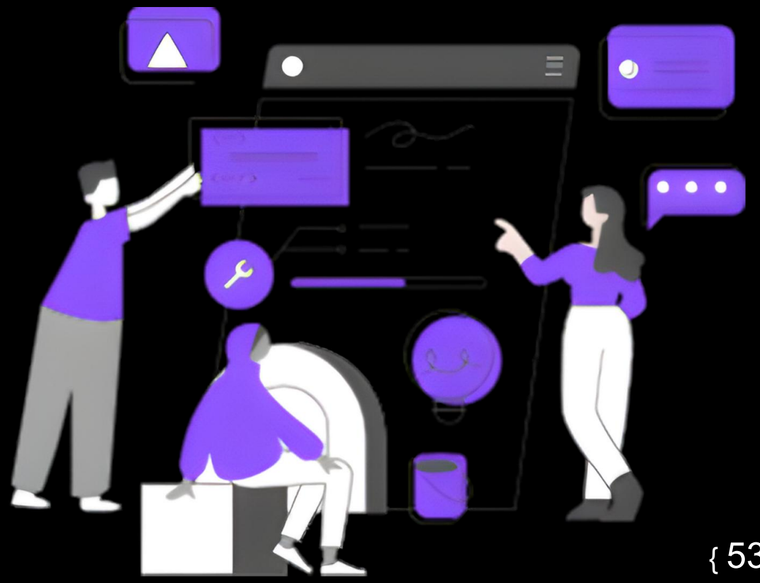
Özyinelemeli Fonksiyonlar



*1- **Base Case** (Temel Durum)*

Durma şartıdır.

Fonksiyonun kendini çağırmayı bıraktığı durumdur.

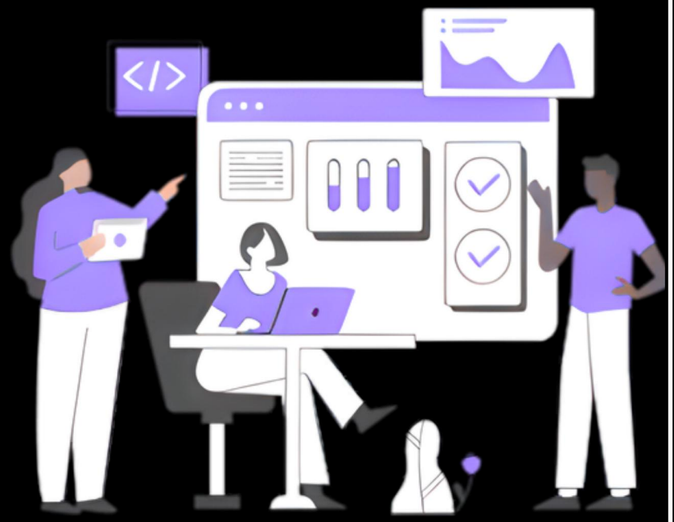


02

Özyinelemeli Fonksiyonlar

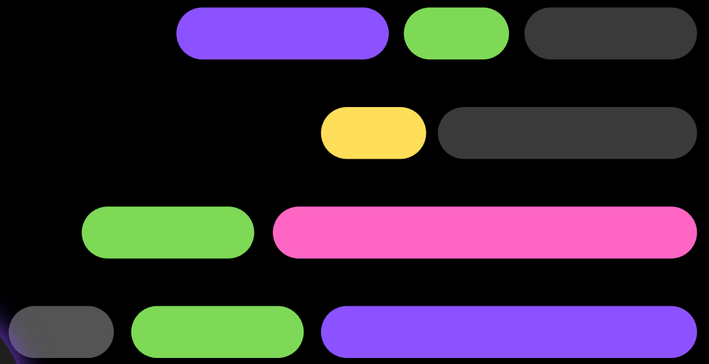
*1- **Recursive Case** (Özyinelemeli Durum)*

Fonksiyonun kendini tekrar çağırdığı durumdur.



02

Özyinelemeli Fonksiyonlar

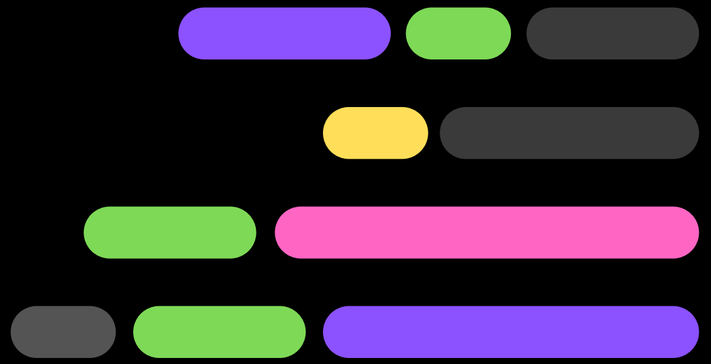


```python
def geri_say(n):
    if n == 0:
        print("Bitti")
    else:
        print(n)
        geri_say(n-1)

geri_say(5)
```

02

Özyinelemeli Fonksiyonlar



Base Case

```python
def geri_say(n):
    if n == 0:
        print("Bitti")
    else:
        print(n)
        geri_say(n-1)

geri_say(5)
```

02

Özyinelemeli Fonksiyonlar

Base Case

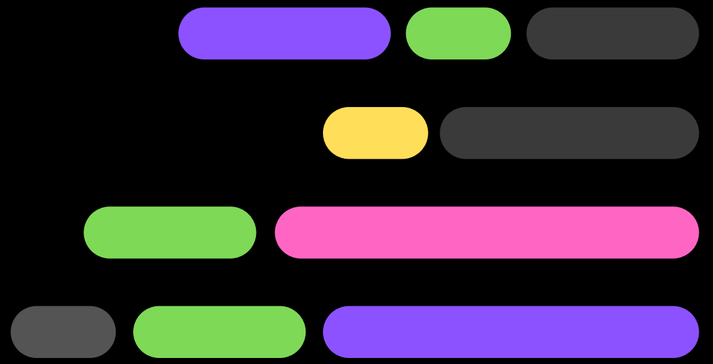
Recursive Case

```python
def geri_say(n):
    if n == 0:
        print("Bitti")
    else:
        print(n)
        geri_say(n-1)

geri_say(5)
```

02

Özyinelemeli Fonksiyonlar



```python
def geri_say(n):
    if n == 0:
        print("Bitti")
    else:
        print(n)
        geri_say(n-1)

geri_say(5)
```

Kod Akışı:

geri_say(5) çağrıldı.

İf'e girmedi, else girdi.

Ekrana 5 yaz

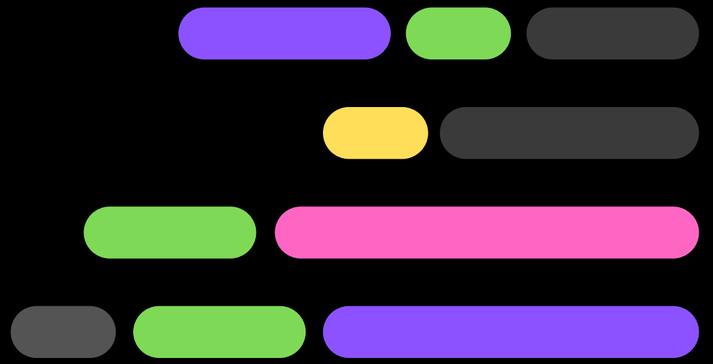
geri_say(4) fonksiyonunu çağır

ÇIKTI :

5

02

Özyinelemeli Fonksiyonlar



```python
def geri_say(n):
    if n == 0:
        print("Bitti")
    else:
        print(n)
        geri_say(n-1)

geri_say(5)
```

Kod Akışı:

geri_say(4) çağrıldı.

İf'e girmedi, else girdi.

Ekrana 4 yaz

geri_say(3) fonksiyonunu çağır

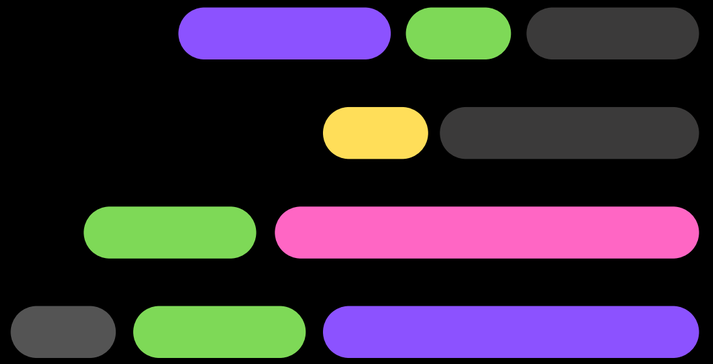
ÇIKTI :

5

4

02

Özyinelemeli Fonksiyonlar



```python
def geri_say(n):
    if n == 0:
        print("Bitti")
    else:
        print(n)
        geri_say(n-1)

geri_say(5)
```

Kod Akışı:

geri_say(3) çağrıldı.

İf'e girmedi, else girdi.

Ekrana 3 yaz

geri_say(2) fonksiyonunu çağır

ÇIKTI :

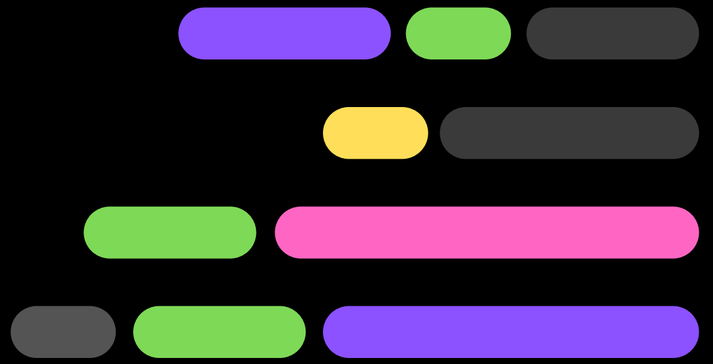
5

4

3

02

Özyinelemeli Fonksiyonlar



```python
def geri_say(n):
    if n == 0:
        print("Bitti")
    else:
        print(n)
        geri_say(n-1)

geri_say(5)
```

Kod Akışı:

geri_say(2) çağrıldı.

İf'e girmedi, else girdi.

Ekrana 2 yaz

geri_say(1) fonksiyonunu çağır

ÇIKTI :

5

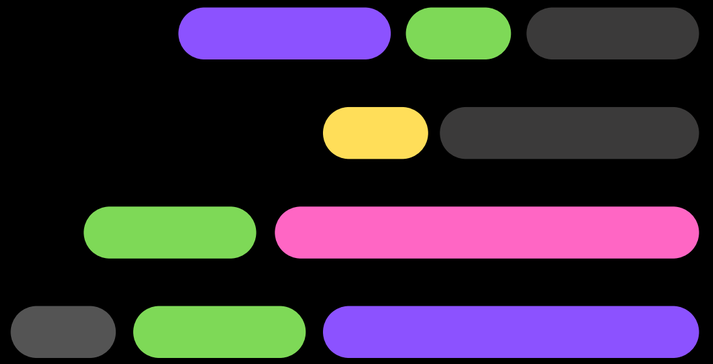
4

3

2

02

Özyinelemeli Fonksiyonlar



```python
def geri_say(n):
    if n == 0:
        print("Bitti")
    else:
        print(n)
        geri_say(n-1)

geri_say(5)
```

Kod Akışı:

geri_say(1) çağrıldı.

İf'e girmedi, else girdi.

Ekrana 1 yaz

geri_say(0) fonksiyonunu çağır

ÇIKTI :

5

4

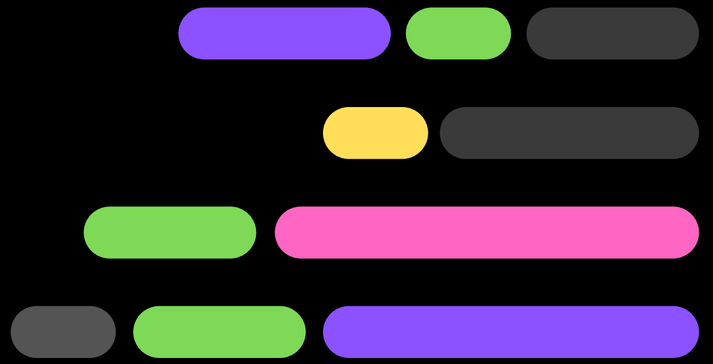
3

2

1

02

Özyinelemeli Fonksiyonlar



```python
def geri_say(n):
    if n == 0:
        print("Bitti")
    else:
        print(n)
        geri_say(n-1)

geri_say(5)
```

Kod Akışı:

geri_say(0) çağrıldı.

İf'e girdi

Ekrana bitti yaz

ÇIKTI :

5

4

3

2

1

Bitti

02

Özyinelemeli Fonksiyonlar

```python
def fonksiyon(n):
    if n == 0:
        return 1
    else:
        return n * fonksiyon(n-1)

print(fonksiyon(5))
```

Yukarıdaki fonksiyon'un amacı ne olabilir ?

02

Özyinelemeli Fonksiyonlar

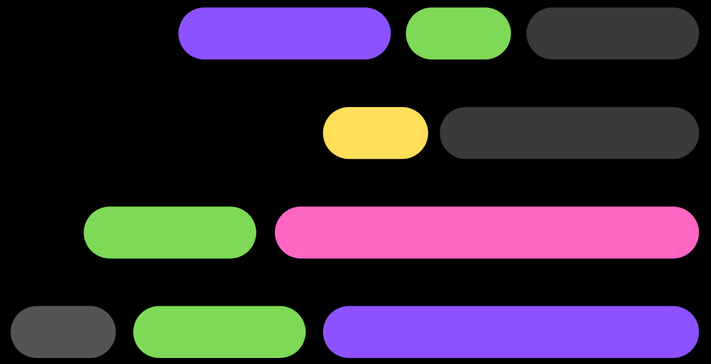
```python
def faktoriyel(n):
    if n == 0:
        return 1
    else:
        return n * faktoriyel(n-1)

print(faktoriyel(5))
```

Base case ve recursive case alanlarını belirtiniz.

02

Özyinelemeli Fonksiyonlar



```python
def faktoriyel(n):
    if n == 0:
        return 1
    else:
        return n * faktoriyel(n-1)

print(faktoriyel(5))
```

$$
\begin{aligned}
&f(5)\\
&= 5 * f(4)\\
&= 5 * (4 * f(3))\\
&= 5 * (4 * (3 * f(2)))\\
&= 5 * (4 * (3 * (2 * f(1))))\\
&= 5 * (4 * (3 * (2 * (1 * f(0)))))\\
&= 5 * (4 * (3 * (2 * (1 * 1))))= 120
\end{aligned}
$$

Uygulama Saati

03

Uygulama Saati

Yandaki şekilde çalışacak bir kargo fiyat hesaplama ve ücret alma uygulaması yazınız.
Kargo ücretini hesaplamak için;
*Yük. * Gen. * Uz. * 0.005 kullanabilirsiniz.*

```
======== KARGO ÜCRETİ HESAPLAMA ========

Yükseklik (cm) : 30
Genişlik (cm)  : 40
Uzunluk (cm)   : 50

Hesaplanan kargo ücreti: 300.0 TL
Alınan para: 200

Eksik ödeme yapıldı! Lütfen yeterli miktarda para ödeyin.
Alınan para: 400

Ödeme tamamlandı!
Para üstü: 100.0 TL
```

Son...