Fonksiyonlar

- Fonksiyonlar herhangi bir işlemi yerine getirmek üzere hazırlanan program parçalarıdır.
- Tekrar yapılması gereken işlemler için hazırlanır.
- print(), input() gibi hazır fonksiyonlar vardır.
- Fonksiyonlar isimleri çağırılarak kullanılır (call).
- Fonksiyonların parametreleri (argümanları) olabilir. print("merhaba") gibi.
- Fonksiyonlar çağırıldıkları yere değer geri döndürebilir. a=input("sayı:") gibi.

- **Fonksiyon yazmak, tanımlamak:**

```
def <fonksiyon adı>([parametre(ler)]):
    <deyim(ler)>
    [return <değer(ler)>]
```

Örnek:

```
def merhaba_yaz():
    print("merhaba")
```

Çağırılan fonksiyon daha önce yazılmış (daha üstte) olmalıdır.

Çağırılması:

```
merhaba_yaz()
```

```
merhaba
```

- **Parametre kullanılması:**

```
def merhaba(n):
    for i in range(n):
        print("merhaba")
```

2 değeri n parametresine aktarılır.

Çağırılması:

```
merhaba(2)
```

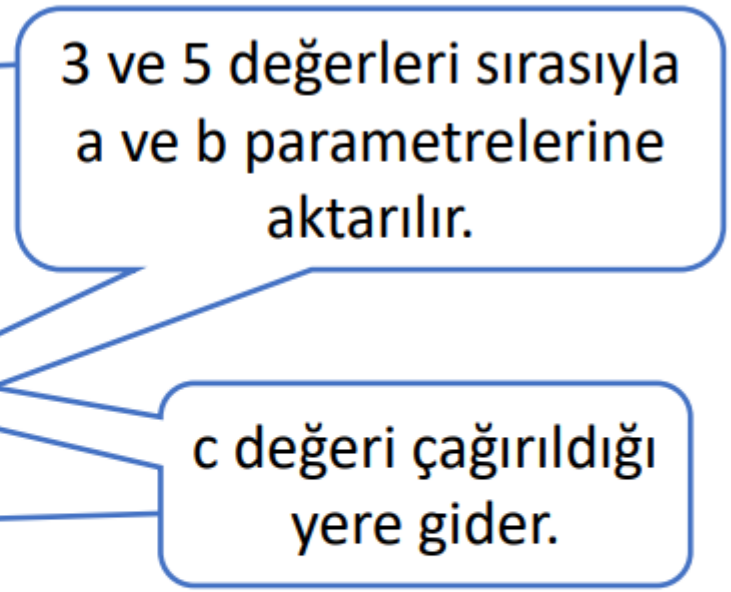
```
merhaba
merhaba
```

- **Değer geri döndürülmesi:**

```
def topla(a,b):
    c=a+b
    return c

Çağırılması:

toplam=topla(3,5)
print(toplam)
```




```
8
```

- **"return" deyimi fonksiyonu sonlandırır.**

```
def topla(sayılar):
    toplam = 0
    for x in sayılar:
        toplam += x
    return toplam
    print("Sayıların toplamı hesaplandı")
print(topla((5, 8, -3, 0, 9)))
```

"return" deyiminden sonra yazıldığı için çalışmaz.

Fonksiyonu çağıran deyim.

```
19
```

- **Fonksiyonun parametreleri, programın diğer kesimlerinde görünmez (erişilmez).**

```
def topla(a,b):
    c=a+b
    return c
toplam=topla(3,5)
print(toplam)
print(a,b)
```

a ve b parametreleri topla fonksiyonu için yereldir.

Fonksiyonu çağıran kesimde a ve b değişkenleri geçerli değildir.

```
8
NameError: name 'a' is not defined
```

- **"return" deyimi birden çok değer geri döndürebilir.**

```
def enb_enk(sayılar):
    enb=max(sayılar)
    enk=min(sayılar)
    return enb,enk
eb,ek=enb_enk((5, 8, -3, 0, 9))
print(eb,ek)
```

iki tane değer geri döndürüyor.

```
9 -3
```

- **Bir fonksiyon başka bir fonksiyonu çağırılabilir.**

```
def enb_2( x, y ):
    if x > y:
        return x
    return y
def enb_3( x, y, z ):
    eb=enb_2( y, z )
    enb=enb_2( x, eb )
    return enb
# Çağır
print(enb_3(3, 6, -5))
```

Bir fonksiyon kendisinden önce yazılmış başka bir fonksiyonu çağırıyor.

```
6
```

- **Parametrelerin varsayılan değeri olabilir.**

```
def topla(sayı1, sayı2=0):
    return (sayı1 + sayı2)
```

```
print(topla(1,2))
```

```
3
```

```
print(topla(2))
```

```
2
```

```
print(topla())
```

```
TypeError: topla() missing 1 required positional argument: 'sayı1'
```

- **Fonksiyon çağırırken parametrelerin adı (keyword) kullanılabilir.**

```
def üçgen_yap(n,ch):
    for i in range(1,n+1):
        print(i*ch)

n=int(input("1-15 arasında bir sayı giriniz:"))
karakter=input("karakter:")
if 1<=n<=15:
    üçgen_yap(n,karakter)
else:
    print("Hatalı veri")
```

Parametre adları kullanılmadı.

```
1-15 arasında bir sayı giriniz:4
karakter:e
e
ee
eee
eeee
```

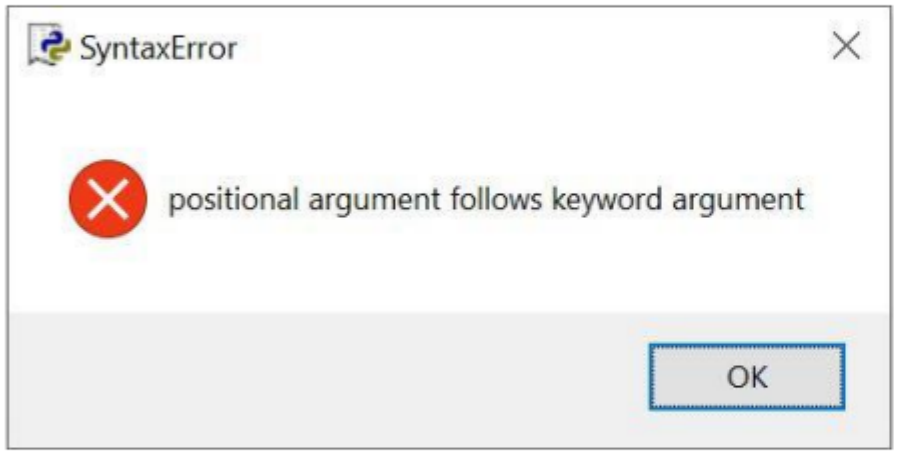
```
def üçgen_yap(n,ch):
    for i in range(1,n+1):
        print(i*ch)

n=int(input("1-15 arasında bir sayı giriniz:"))
karakter=input("karakter:")
if 1<=n<=15:
    üçgen_yap(ch=karakter, n=n)
else:
    print("Hatalı veri")
```

Parametre adları kullanıldı. Sırası önemli değil. parametre n ile değişken n, aynı isme sahip olabilir.

```
1-15 arasında bir sayı giriniz:3
karakter:x
x
xx
xxx
```

üçgen_yap(ch=karakter,n)



Bir parametrenin adı kullanıldığında, varsa sonraki parametrelerde de ad kullanılmalıdır.

üçgen_yap(ch, n=n)

Adı kullanılan parametreler sona yazılabilir.

- **Parametre sayısının değişken olması (önceden belirlenmemesi).**

```
def çarpım(*prm):
    sonuç=1
    for i in prm:
        sonuç *= i
    return sonuç

ç=çarpım(1,2,3)
print(ç)
ç=çarpım()
print(ç)
```

```
6
1
```

* işareti parametrenin çok değer alabileceğini gösterir.

```
def çarpım(*prm):
    print(type(prm))
    print(prm)

çarpım(1,2,3)
```

* işaretli parametre demet (tuple) yapısına dönüşür.

```
<class 'tuple'>
(1, 2, 3)
```

- **Fonksiyon yazılmadan önce kullanılan değişkenler ve "global" kelimesi.**

```
sayı=15

def fonk():
    print(sayı)

fonk()
```

```
15
```

Fonksiyon yazılmadan önce değer alan değişkenler fonksiyonun içinde görünür.

```
sayı=15

def fonk():
    sayı=20
    print(sayı)

fonk()
print(sayı)
```

Fonksiyon içinde değer aktarıldığı için sadece fonksiyon içinde geçerlidir. Fonksiyondan önce değer aktarılan "sayı" değişkeninden farklıdır.

```
20
15
```

```
sayı=15

def fonk():
    sayı=20*sayı
    print(sayı)

fonk()
```

Fonksiyon yazılmadan önce değer alan değişkenlerin değeri fonksiyonun içinde değiştirilemez (nonlocal değişken).

```
UnboundLocalError: local variable 'sayı' referenced before assignment
```

```
sayı=15

def fonk():
    global sayı
    sayı=sayı*2
    print(sayı)

fonk()
print(sayı)
```

Fonksiyon içinde kullanılan "sayı" değişkeninin yerel olmadığını belirtir.

```
30
30
```

```
sayı=15

def fonk(sayı):
    sayı=sayı*2
    print(sayı)

fonk(20)
print(sayı)
```

Parametreler yereldir (local).

```
40
15
```