**Fonksiyonların Tanımlanması**

Fonksiyon,belirli sayıda verileri kullanarak bunları işleyen ve bir sonuç üreten komut grubudur.Fonksiyonlar programlamada kullanacağımız ve sürekli tekrar eden komutların kullanımını daha da kolaylaştırmak için oluşturacağımız yapılardır.

Bu tanımlamaları yaptıktan sonra şu şekilde bir açıklama yapabiliriz. Elimizde bir kod kümesi var ve biz bunu programımız için de tekrar tekrar kullanıyoruz. Bu noktada fonksiyonlar devreye giriyor ve bu kod kümesini tekrar tekrar yazmak yerine bir fonksiyon haline çeviriyoruz. İstediğimiz yerde bu fonksiyonu çağırıyoruz.

Bu kodu paketleyerek tekrar tekrar kullanmamızı sağlayan yaklaşımlardan biri "fonksiyonlar"dır. Bir fonksiyon, tekrar kullanılabilen kod parçacığıdır. Kendimiz fonksiyon yazabileceğimiz gibi önceden yazılmış ve kullanıma hazır fonksiyonları da kullanabiliriz.

Fonksiyon tanımlamanın yapısı şu şekildedir.

```
def fonksiyon  adı(parametre1,parametre2..... (opsiyonel)):
        # Fonksiyon bloğu
         Yapılacak işlemler
        # dönüş değeri – Opsiyonel
```

Ör: ……..

```
def selamla():
        print("Selam arkadaşlar...")
        print("Nasılsınız?")
selamla()
```

Çıktısı:

Selam arkadaşlar...
Nasılsınız?

Ör: ……..

```
def selam(isim): # isim değişkenimiz burada parametre olmaktadır
    print("Merhaba:",isim)
selam("Kemal")
```

Çıktısı: Merhaba: Kemal

Ör: ……..

```
def toplama(a,b,c):
        print("Toplamları:",a+b+c)
toplama(3,4,5)
toplama(4,9,40)
```

Çıktısı:

Toplamları: 12
Toplamları: 53

Ör:……..

```
def karealanihesapla(n):
    print("karenin alanı=",n*n)
def ucgeninalani(a,h):
    print("üçgenin alanı=",(a*h)/2)
print("program başladı.")
ucgeninalani(4,6)
karealanihesapla(8)
```

Çıktısı:

program başladı.
ücgenin alanı= 12.0
karenin alanı= 64

**RETURN DEYİMİ:**

Yukarıdaki fonksiyonlar sadece ekrana **print** ile değer yazdırıyor. Anca k bu fonksiyonlar yaptıkları işlemler bize herhangi bir değer vermiyor. Ancak biz programlarımızda bir fonksiyon sonucunda elde edilen değerl eri alıp programlarımızın bambaşka yerlerinde kullanmak isteyebiliriz. **Return** ifadesi fonksiyonun işlemi bittikten sonra **çağrıldığı yere** değer döndürmesi anlamı taşır. Böylelikle, fonksiyonda aldığımız değeri bir de ğişkende depolayabilir ve değeri programın başka yerlerinde kullanabilir iz.

Ör:……..

```
def toplama(a,b,c): # Birinci fonksiyon
    print("Toplamları",a+b+c)

def ikiyleçarp(a): # İkinci fonksiyon
    print("2 ile çarpılmış hali", a * 2)

toplam = toplama(3,4,5)
ikiyleçarp(toplam)
```

Çıktısı: Hata verir.

Programımızı Şöyle yazsaydık:

```
def toplama(a,b,c):
    return a+b+c
def ikiyle_çarp(a):
    return a*2

toplam = toplama(3,4,5)
print(ikiyle_çarp(toplam))
```

Çıktısı: 24

Ör: ……..

```
def ucleçarp(a):
    print("1.fonksiyon çalıştı")
    return a*3
def ikiyletopla(a):
    print("2.fonksiyon çalıştı")
    return a + 2
def dordebol(a):
    print("3.fonksiyon çalıştı")
    return a / 4
print(dordebol(ikiyletopla(uclecarp(4))))
```

Çıktısı:

1.fonksiyon çalıştı
2.fonksiyon çalıştı
3.fonksiyon çalıştı
3.5

**MODÜL MANTIĞI**

**Pythonda aslında her bir dosya bir** modüldür **ve her bir modül içinde fonksiyonlar, sınıflar ve objeler barındırır. Biz de bu modülleri programımıza** dahil ederek **içindeki fonksiyonlardan, sınıflardan ve objelerden faydalanabiliriz.**

**math Modülü Kullanımı**

**YÖNTEM1 - import modül_adı**

**Ör: import math** *# Modülü içeri aktarıyoruz.Artık bu modülün tüm fon ksiyonlarını kullanabiliriz.*

Peki bu içeri aktarma yöntemiyle **math** modülünün herhangi bir fonksiyonunu nasıl kullanacağız.

modül_adı.fonksiyonadı()

Ör: ……..

```
import math
math.factorial(5)
```

Çıktısı: 120

**YÖNTEM2 - from modül_adı import**

```
from math import factorial
floor(3.5)
```

Çıktısı: 120

Ör: ……..

```
from math import factorial,floor,ceil
floor(3.5)  (Kendisinden önceki en büyük tamsayıyı veren fonksiyon)
factorial(3)
ceil(3.5) (Kendisinden sonraki en küçük tamsayıyı veren fonksiyon)
```

Çıktısı:

3
6
4

NOT: Bir de **from math import** * şeklinde bir kullanım vardır. Buda math modülündeki tüm fonksiyonları ekle demektir.

```
from math import *
floor(8.9)  (Kendisinden önceki en büyük tamsayıyı veren fonksiyon)
factorial(4)  (# 4*3*2*1 demektir.)
ceil(7.2) (Kendisinden sonraki en küçük tamsayıyı veren fonksiyon)
```

Çıktısı:

8
24
8

ÖRNEKLER:

**pow (Üs Alma) :**
#Verilen iki sayıdan ilk sayının ikinci sayı kuvvetini bulur.

```
>>> math.pow(3,2)
9.0
```
---

**sqrt (Karekök) :**
#Verilen sayının karekökünü bulur.

```
>>> math.sqrt(16)
4.0
```

**Ceil(): Parantez içerisinde verilen sayıdan büyük olan,en küçük tamsayıyı verir.**

```
>>> math.ceil(5.6)
6
```

floor(): **Parantez içerisinde verilen sayıdan küçük olan,en büyük tamsayıyı verir.**

```
>>> math.ceil(5.6)
5
```

---

**sleep() Fonksiyonu**
time modülünün sleep() fonksiyonu programımızı istediğimiz süre kadar durdurmamızı sağlar. Aldığı argüman saniye cinsindendir.

```
import time
time.sleep(2 )
```

Bu kod programımızı 2 saniyeliğine durduracaktır.

---

**clock()**: Programın başladığı andan itibaren geçen süreyi saniye olarak hesaplar.

---

**Random:** Modülün bu metodunu kullandığımız zaman 0 ile 1 arasında rastgele bir sayı üretilir.

```
>>>import random
>>>random.random()
0.8759562487514273
```

---

**Randint:** Random ile ürettiğimiz sayı float tipinde bir sayı idi. Biz integer tipinde bir sayı istiyorsak bu durumda randint metodunu kullanabiliriz. Bu metot kullanılırken başlangıç ve bitiş aralık değerleri verilir; ancak bu durumda bitiş değeri de rastgele sayı olarak tutabilir. Yani [başlangıç, bitiş] aralığı kullanılır.

```
    >>>random.randint(1, 3)
    3
import random
print(random.randint(4,46))   # 4 < = a < = 46
```

**(4,5,6,…….45,45,46 değerlerinden birini verir.)**

---

**Randrange:** Başlangıç, bitiş ve artış miktarını belirleyerek rastgele sayı üretmek istiyorsak bu durumda kullanabileceğimiz metot ise randrange metodudur.

```
import random
print(random.randrange(4,46))   (# 4 < = a < 46
```

**(4,5,6,…….45,45 değerlerinden birini verir.)**

**uniform(başlangıç,bitiş)**

```
import random
print(random.uniform(1, 4.5)) ( 1.0 < = a  < 4.5 )
```

(1.0, 1,1 ,……., 4.4 değerlerinden birini alır.)

**choice(list):** Verdiğiniz bir liste içinden rastgele bir değer seçer.

```
import random
liste=["ali","osman","murtaza","ayşe","kaya","mehmet","mustafa"]
print(random.choice(liste))
murtaza
```

ÖRNEKLER:

```
def toplama(a,b,c):
    return a+b+c
def ikiylecarp(a):
    return a*2
t=toplama(1,2,3)
print(ikiylecarp(t))
```

Çıktısı:12

```
def dordebol(a):
    print("1.fonksiyon çalıştı")
    return  a/4
def uclecarp(a):
    print("2.fonksiyon çalıştı")
    return  a*3

def ikicikar(a):
    print("3.fonksiyon çalıştı")
    return  a-2
print(ikicikar(dordebol(uclecarp(8))))
```

Çıktısı:

```
2.fonksiyon çalıştı
1.fonksiyon çalıştı
3.fonksiyon çalıştı
4.0
```

……..

**Ör: 10'a kadar sayma**

```
def say():
    for i in range(1, 11):
        print(i, end=" ")
say()
```

**Çıktısı:  1 2 3 4 5 6 7 8 9 10**

……..

Ör: Sayısal Loto Programı

```
import random
    for i in range(6):   # Rastgele 6 adet sayı üretilecek
        print(random.randrange(1, 50), end=" ")   # 1 ile 50 arasında
```

(NOT:Burada end komutu çıktıların yan yana yazılmasını sağlamak için kullanılmıştır.)
Çıktı: 35, 41, 3, 45, 12, 8

……..

Ör: Bilgi girilmeden önce 5 saniye bekleten program.

```
import time
time.sleep(5)
ad=input("İsminizi Giriniz: ")
```

……..

Ör: İç içe karekök Fonksiyonu

```
from math import sqrt
y = sqrt(sqrt(256))
print(y)
```

Çıktı: 4

……..

```
from math import sqrt
x=9
b= 2 * sqrt(x + 16) - 4
print(b)
```

Çıktısı: 6.0

……..

```
def sayininonkati(a):
    print("sayınızın 10 katı=",a*10)
def kareninalani(b):
    print("karenizin alanı=",b*b)

kareninalani(3)
sayininonkati(4)
print("proram çalışıyor...")
sayininonkati(6)
kareninalani(2.4)
```

ÇIKTISI:

```
karenizin alanı= 9
sayınızın 10 katı= 40
proram çalışıyor...
sayınızın 10 katı= 60
karenizin alanı= 5.76
```