PYTHON

İÇERİK

- ✓ **Fonksiyonlar**

KAZANIMLAR

- ✓ Fonksiyon nedir ve nasıl kullanılır öğrenir
- ✓ Fonksiyon çeşitleri nelerdir kavrar

Fonksiyon

Fonksiyonların görevi, karmaşık işlemleri bir araya toplayarak, bu işlemleri tek adımda yapmamızı sağlamaktır. Fonksiyonlar çoğu zaman, yapmak istediğimiz işlemler için bir şablon vazifesi görür. Fonksiyonları kullanarak, bir veya birkaç adımdan oluşan işlemleri tek bir isim altında toplayabiliriz. Python'daki 'fonksiyon' kavramı başka programlama dillerinde 'rutin' veya 'prosedür' olarak adlandırılır. Gerçekten de fonksiyonlar rutin olarak tekrar edilen görevleri veya prosedürleri tek bir ad/çatı altında toplayan araçlardır.

Şu ana kadar hep bir fonksiyon kavramından bahsettik print(), len(), type() bunlar bizim için fonksiyon örnekleridir.

Buraya kadar öğrendiğimiz fonksiyonlardan yola çıkararak şunları söyleyebiliriz:

1. Her fonksiyonun bir adı bulunur ve fonksiyonlar sahip oldukları bu adlarla anılır. (print fonksiyonu, type fonksiyonu, input fonksiyonu, len fonksiyonu vb.)
2. Şekil olarak, her fonksiyonun isminin yanında birer parantez işareti bulunur. (print(), input(), len() vb.)
3. Bu parantez işaretlerinin içine, fonksiyonlara işlevsellik kazandıran bazı parametreler yazılır. print("Merhaba Zalim Dünya!"), len("kahramanmaraş") vb.)
4. Fonksiyonlar farklı sayıda parametre alabilir. Örneğin print() fonksiyonu toplam 256 adet parametre alabilirken, input() fonksiyonu yalnızca tek bir parametre alır.
5. Fonksiyonların isimli ve isimsiz parametreleri vardır. print() fonksiyonundaki sep, end ve file parametreleri isimli parametrelere örnekken, mesela print("Merhaba Dünya!") kodunda Merhaba Dünya! parametresi isimsiz bir parametredir. Aynı şekilde input("Adınız: ") gibi bir kodda Adınız: parametresi isimsiz bir parametredir.
6. Fonksiyonların, isimli ve isimsiz parametreleri dışında, bir de varsayılan değerli parametreleri vardır. Örneğin print() fonksiyonunun sep, end ve file parametreleri varsayılan değerli parametrelere birer örnektir. Eğer bir parametrenin varsayılan bir değeri varsa, o parametreye herhangi bir değer vermeden de fonksiyonu kullanabiliriz. Python bu parametrelere, belirli değerleri öntanımlı olarak kendisi atayacaktır. Tabii eğer istersek, varsayılan değerli parametrelere kendimiz de başka birtakım değerler verebiliriz.

En basit ifade ile fonksiyonlar

- ✓ Bir işlev, yalnızca çağrıldığında çalışan bir kod bloğudur.
- ✓ Parametreler olarak bilinen verileri bir işleve geçirebilirsiniz.
- ✓ Bir işlev, sonuç olarak veri döndürebilir.

Fonksiyon Oluşturma

Python da def anahtar sözcüğü kullanılarak bir fonksiyon tanımlanır :

```
def fonksiyon():

  print("Fonsiyondan merhaba")
```

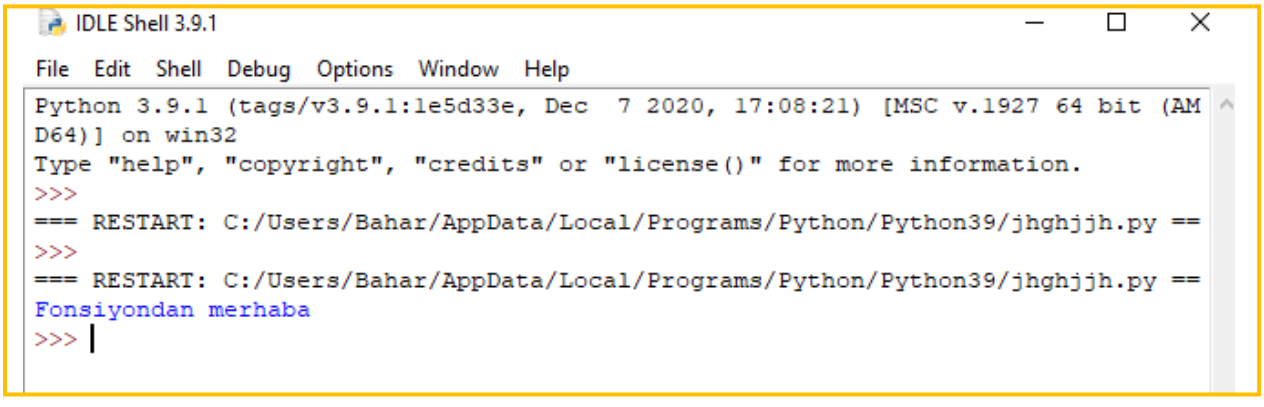
Fonksiyonumuzu def sözcüğü ile tanımladık fonksiyonumuzun ismini kendimiz verdik. Fonksiyondan işlev olarak print() komutu ile ekrana Fonksiyondan merhaba yazdırmasını istedik. Kodu çalıştırdık fakat ekrana herhangi bir şey gelmedi ve hata da vermedi. Çünkü bizim fonksiyonumuzu çağırmadık Bir fonksiyonu çağırmak için fonksiyon adını ve ardından parantezi kullanılır:

```
def fonksiyon():

  print("Merhaba bu benim ilk fonksyonum")

fonksiyon()
```

Kodumuzu şimdi çalıştırdığımızda ekrana gelen çıktı görseldeki gibi olacaktır.



Kullanıcıdan iki sayı değeri alan ve bunları yazan bir fonksiyon tanımlayalım.

```
def Toplama():
    ilksayı=int(input("Birinci sayıyı giriniz:"))
    ikincisayı=int(input("İkinci sayıyı giriniz:"))
    topla=int(ilksayı+ikincisayı)
    print("Toplam:",topla)
Toplama()
```

Bu programda önce Toplama() isimli fonksiyonumuzu oluşturduk. Ardından ilksayi ve ikincisayi yı input fonksiyonu ile kullanıcıdan aldık. Sayıları toplama işlemine sokup print fonksiyonuyla toplamı ekrana yazdırdık. Aslında burada fonksiyon içerisinde başka bir fonksiyonun kullanımını da görmüş olduk. Programımızda parantezlerin dışında yer alan int ifadeleri ise kullanıcının girdiği sayıların Integer türünde olduğunu kontrol ediyor. Eğer bir tam sayı yerine 3.5 gibi bir ondalıklı sayı veya “Ahmet” gibi bir kelime girersek program hata verecektir.

Şimdi matematiksel bir fonksiyon yazalım. Yarıçapı girilen bir dairenin alanını hesaplayan uygulama yapalım. Normalde bunu $\pi r.\, r$ fonksiyonuyla yapıyoruz. Girilen yarıçapın karesini alıyoruz ardından bunu matematiksel bir sabit olan π (pi) sayısıyla çarpıyoruz. Fonksiyonumuzu yazalım. Adı alan_hesapla olsun.

```
def alan_hesapla():
    pi=3.14
    r=float(input("yarıçap:"))
    alan=r*r*pi
    print(alan)
alan_hesapla()
```

Önce alan_hesapla() isimli fonksiyonumuzu oluşturduk. Fakat henüz boş. İçine önce pi=3.14 diyerek bir değişken ekledik. Pi sayısının değerini 3.14 kabul edeceğimizi söylemiş olduk. Ardından r= float(input("Yarıçap giriniz:")) satırıyla kullanıcıdan float değişken türünde bir yarıçap girilmesini istedik. Eğer Integer olarak bu kısmı alsaydık kesirli bir sayı olan 3.14 ile çarparken sorun çıkacaktı. Program hata verecekti. Ardından alan formülümüzü yazdı. r*r*pi ile alan hesapladık. Print fonksiyonuyla bulunan alanı ekrana yazdık.

Parametre

Şimdiye kadar olan fonksiyonlarda, fonksiyonumuz herhangi bir parametre almıyordu. Yani alan_hesapla() şeklinde doğrudan çalıştırıyorduk. Peki buradaki parantezlerin içi hep boş mu kalacak ? Bir değer alamaz mı ? Yani alan_hesapla(4) diyemez miyiz ? Böyle olsa daha güzel olur bence. Hemen 4 ün alanı hesaplanır.

```
def alan_hesapla(r):
    pi=3.14
    alan=r*r*pi
    print(alan)
alan_hesapla(4)
print("şimid yarıçapı 5 yapalım")
alan_hesapla(5)
```

İlk başta def alan_hesapla() yerine def alan_hesapla(r) dedik. Yani Python'a dedik ki "Bak sevgili Python, fonksiyonumun içerisinde bir tane r değişkeni var. Ben fonksiyonu çalıştırırken bu r değişkenini sana söyleyeceğim. Sen içerideki kodlarda bunu kullan." Bunu yaptıktan sonra fonksiyonu artık alan_hesapla() şeklinde değil de alan_hesapla(4) , alan_hesapla(5) gibi ifadelerle çağırdık. 4 için 50.24 5 içinse 78.5 sonucunu bize verdi. İşte bir parametreyle fonksiyon kullanmak bu kadar basit.

return komutu

Bir fonksiyonun bir değer döndürmesine izin vermek için return ifadesi kullanılır

```
def katları(x):

  return 5 * x

print(katları(3))

print(katları(5))

print(katları(9))
```

Kelimeler ve sayıları beraber kullanalım. Fonksiyonumuz birkaç parametre alsın. Şöyle bir senaryo düşünelim. Mert isminde çalışkan bir öğrencimiz var. İlk sınavdan 100 almış fakat ikinci sınavı biraz kötü geçmiş 70 almış. Sisteme bu bilgileri girerek , not ortalamasını hesaplayacağız. Fonksiyonumuz 3 parametre alacak. İlki “Mert” ikincisi 100 ve üçüncüsü 70 olacak.

```
def not_hesapla(isim,not1,not2):

    print(isim,"isimli öğrencinin notu:")

    ortalama=int((not1+not2)/2)

    print("Ortalama:",ortalama)

not_hesapla("Mert",100,70)
```

Görüldüğü gibi programımız isim kısmını Mert isimli öğrencinin notu ifadesi içerisinde kullandı. Notlardan ise bir ortalama alarak Ortalama: ifadesinin karşısında sonucu hesapladı.

Sıra Sizde

Adlı bir işlev oluşturun my_function. Boşluğu uygun şekilde tamamlayınız.

```
[ ]:
    print ("Bir işlevden merhaba")
```

cevap: def my_function:

Şimdi bu zamana kadar öğrendiğimiz modüller fonksiyonlar ve döngüleri kullanarak bir örnek yapalım.

```
import turtle
t=turtle.Pen()
def şekilçiz(kenarsayısı):
    dışaçı=360/kenarsayısı
    for x in range(kenarsayısı):
        t.forward(100)
        t.right(dışaçı)
şekilçiz(3)
t.clear()
şekilçiz(4)
t.clear()
şekilçiz(5)
t.clear()
şekilçiz(6)
t.clear()
şekilçiz(7)
t.clear()
```

Ödev

Turtle modülünü kullanarak öğrencilerden merdiven çizmeleri istenir. Bunu yaparken fonksiyonlar ve döngüler kullanılmalıdır