# Python Ögren

May 7, 2025

## 1 Yazarlar

### 1.1 Ege Altay

## 2 Telif Hakkı

## 3 ————-

## 4 Python nedir?

Python nesne yönelimli, açık kaynak kodlu çok amaçlı bir programlama dilidir. Yaratıcısı Guido van Rossum'dur. Kullanım amacına yönelik birçok kütüphanesi (library) bulunmaktadır. Birçok işletme tarafından tercih edilmektedir. Neredeyse her türlü işletim sisteminde çalışabilir. Örneğin:

- Windows
- Linux
- MacOS

Öğrenmesi diğer dilleri göre daha kolaydır. İlk defa programlama dili öğrenmek istiyorsanız idealdır.

- Python'ın resmi web sitesi: https://www.python.org/

### 4.1 Bazı python kütüphaneleri

#### 4.1.1 Web tasarımı

- Django
- Flask

#### 4.1.2 Matematik kütüphaneleri

- Matplotlib
- numpy
- SciPy

#### 4.1.3 Yapay zeka kütüphaneleri

- PyTorch
- Tensorflow
- Scikit-learn

#### 4.1.4 Veri tabanı kütüphaneleri

- Pandas
- SQLAlchemy

#### 4.1.5 Oyun kütüphaneleri

- PyGame

#### 4.1.6 Web'den veri toplama (Web scraping/Web crawling) kütüphaneleri

- Scrapy
- BeautifulSoup

#### 4.1.7 Web otomasyon ve test kütüphaneleri

- Selenium

#### 4.1.8 Resim düzenleme ve manipülasyonu

- Pillow
- Bir gelenek olarak aşağıdaki kodu çalıştırarak başlayalım

[1]:
```
print("Merhaba Dünya!")
```

```
Merhaba Dünya!
```

## 5 Pythondaki Basit Veri Tipleri

Her programlama dilinde olduğu gibi python'da da bazı veri tipleri bulunmaktadır. Veri tipi dediğimiz şey belirli türdeki sayı, yazı, Liste gibi değerlerdir. Merak etmeyin bunlara sıklıkla değineceğiz. Bu verileri değişken dediğimiz şeylere atarız.

- Önemli NOT: değişken isimleri boşluk içeremez ve bir sayıyla başlaymaz. Bu değişkenleri adlandırırken türkçe karakter kullanmamaya çalışmayız. Çünkü kod bizim bilgisayarımızda çalışsa bile diğer bilgisayarlarda çalışmayabilir.

**Hatalı isimlendirme örnekleri** benim adim= "Ege" (Boşluk karakter kullanımı)

yaş = 24 (Türkçe karakter kullanımı)

1metre_kac_cm = 100 (Değişken adı sayıyla başlayamaz)

**Doğru isimlendirme örnekleri** benim_adim= "Ege"

yas = 24

bir_metre_kac_cm = 100

| Veri tipi | Kısaltması ya da tam adı |
|---|---|
| Tam sayı | int |

| Veri tipi | Kısaltması ya da tam adı |
|---|---|
| Ondalık sayı | float |
| Doğru/Yanlış | bool |
| Metin | str |
| Liste | list |
| Küme | set |
| Sözlük | dict |

### 5.1 Tam sayılar (intiger / int)

İlk veri tipimiz intiger (int) dediğimiz tam sayılar olacak. Bu veri tiplerini değişken dediğimiz şeylere atayacağız. Bunu yapmak için şu formulü izleyeceğiz.

- DEĞİŞKEN_ADI = VERİ_DEĞERİ

Bunu yaparken de print dediğimiz bir fonksiyonu kullanarak bunu ekrana yazdıracağız. Bu işleme çoka takılmayın ileride bunu detaylandıracağız Hadi hızlıca aşağıdaki örneğe geçelim.

[2]:
```
yas = 24
print("Yaşım:",yas)


siniftaki_kisi_sayisi = 10
print("Sınıfta kaç kişi var:",siniftaki_kisi_sayisi)

turkiye_nufusu = 85500000
print("Türkiye nüfusu:",turkiye_nufusu)
```

```
Yaşım: 24
Sınıfta kaç kişi var: 10
Türkiye nüfusu: 85500000
```

Gördüğünüz üzere bu sayılar adında geçtiği gibi tam sayı olmak zorunda virgüllü olamazlar. Fakat bazen bu sayılar büyük olunca ayırmak içine arasına "_" bu işareti ekleyebiliriz. Tıpkı bazı büyük sayıları yazarken 3 basamak sonra "." koymak gibi. Örneğin 85.500.000 böylece bu sayıları okumak daha kolay olur.

[3]:
```
turkiye_nufusu = 85_500_000
print("Türkiye nüfusu:",turkiye_nufusu)
```

```
Türkiye nüfusu: 85500000
```

### 5.2 Ondalık sayılar (float)

Diğer veri tipimiz float dediğimiz ondalık sayılar olacak. Bu veri tiplerini değişken dediğimiz şeylere atayacağız. Bunu yapmak için şu formulü izleyeceğiz.

- DEĞİŞKEN_ADI = VERİ_DEĞERİ

Verinin değeri içinde "." olmak zorundadır.

[4]:
```
pi_sayisi = 3.141592
print("Pi sayısı:",pi_sayisi)

boyum_kac_metre = 1.85
print("Metre cinsinden boy uzunluğum:",boyum_kac_metre)


deneme_degeri = 2.0
print("Deneme değeri:",deneme_degeri)
```

```
Pi sayısı: 3.141592
Metre cinsinden boy uzunluğum: 1.85
Deneme değeri: 2.0
```

Yukarıdaki örnekte gördüğünüz gibi bazı tam sayıları da ondalık sayı olarak ifade edebiliyoruz. Örneğin 4 yazarsak tam sayı 4.0 yazarsak ondalık sayı oluyor.

### 5.3 Metin (string / str)

Diğer bir veri tipimiz string (str) dediğimiz metin verisi olacak. Formülü tam sayılar gibi ama bu sefer metninönüne ve başına çift tırnak veya tek tırnak işareti ekliyoruz.

- DEĞİŞKEN_ADI = "METİN"

[5]:
```
isim = "Ege"
print("İsim:",isim)


en_sevdigi_renk = "mavi"
print("En sevdiği renk:",en_sevdigi_renk)

soyad = 'Altay'
print("Soyad:",soyad)
```

```
İsim: Ege
En sevdiği renk: mavi
Soyad: Altay
```

### 5.4 Boolean değerler (bool)

Bool dediğimiz değerler doğru veya yanlış ifadelerdir. Merak etmeyin bunun da bir kullanım alanı var. True ingilizce doğru, False ise yanlış demektir.

- DEĞİŞKEN_ADI = True
- DEĞİŞKEN_ADI = False

Operatörler konusunda bu değere tekrar değineceğiz

[6]:
```
isik_acik_mi = True
print("Işık açık mı?",isik_acik_mi)
```

```
kapi_kapali_mi = False
print("Kapı kapalı mı?",kapi_kapali_mi)
```

```
Işık açık mı? True
Kapı kapalı mı? False
```

- Aynı zamanda "1" sayısı bazı durumlarda "True" değerine tekabül etmketedir. Aynı şekilde de "0" sayısı da bazı durumlarda "False" değerine tekabül etmketedir.

## 6 Aritmetik Operatörler (İşleçler)

Aslında bu bölüm oldukça basit. Aritmetik işlemleri ve bir çok işlemi daha bu operatörler aracılığıyla yapıyoruz.

| Operatör | İşlevi |
|---|---|
| + | Toplama |
| - | Çıkarma |
| * | Çarpma |
| / | Bölme |
| % | Mod alma |
| ** | Kuvvet alma |
| // | Tam bölme (Ondalık kısmı atarak bölme) |

Aşağıda örnekleri bulabilirsiniz

[7]:
```
x = 2
print("X'in ilk değeri:",x)

x = x*2
print("X'in 2 ile çarpımı:",x)

y = 16
print("Y'in ilk değeri:",x)

y = y / 4
print("Y'nin 4'e bölümü:",y)
```

```
X'in ilk değeri: 2
X'in 2 ile çarpımı: 4
Y'in ilk değeri: 4
Y'nin 4'e bölümü: 4.0
```

- Önemli Not: Bir değişkenin değerini tanımladıktan sonra; örneğin z=4 diyelim, z= z/4 yazdığınızda z'nin değerini kalıcı olarak değiştirirsiniz.

[8]:
```
y = 16
print("İlk değer:", y)
```

```
print("4'le modu:", y%4)

print("İkinci kuvveti:", y**2)

print("5'le tam bölümü:", y//5)
```

```
İlk değer: 16
4'le modu: 0
İkinci kuvveti: 256
5'le tam bölümü: 3
```

[9]:
```
sayi_bir = 12.6
print("1. sayı=", sayi_bir)

sayi_iki = 3.15
print("1. sayı=", sayi_iki)

print("İki sayının bölümü:", sayi_bir/sayi_iki)
```

```
1. sayı= 12.6
1. sayı= 3.15
İki sayının bölümü: 4.0
```

### 6.1 String değerleriyle bazı işlemler

[10]:
```
selamlama = "Merhaba "
isim = "Ege"

metin = selamlama + isim

print(metin)

giris = "sınıfımızdaki kişi sayısı:"
sayi = "15"

print(giris+sayi)
```

```
Merhaba Ege
sınıfımızdaki kişi sayısı:15
```

- "+" işareti iki string'i birbiri ile toplar
- Üstteki "sayi" değişkeni aslında bir string. Çift tırnak işaretleri arasında olduğuna dikkat edin.

[11]:
```
metin1 = "A"
metin1 = metin1 * 6

print("metin1=", metin1)
```

```
metin2 = "ABC"
metin2 = metin2*2

print("metin2=", metin2)
```

```
metin1= AAAAAA
metin2= ABCABC
```

- "*" işareti iki string'i o sayıda yineler.

# Yorum satırları (Comment Lines)

Yorum satırları şu ifadeyle başlar # bu ifadenin sağ yanına yazılan kelimeler işleme alınmaz. Bunlara yorum satırları denir

[12]:
```
# Örnek 1
sayim = 12
# ege = 13
print(sayim)
```

```
12
```

## 6.2 Bazı anlamsız operatör kullanımları:

ad = "Ege" soyad = "Altay"

print(ad/soyad)

Bir metni diğer bir metine bölemezsiniz. Kod hata verir. Aynı şekilde bir metni de sayıya bölemezsiniz.

siniftaki_kisi = "12" erkek_orani = 0.6

print("Sınıftaki erkek sayisi:") print(siniftaki_kisi*erkek_orani)

### 6.2.1 Kısa bir hatırlatma

Bir sayının modu demek o sayının istenen sayıya bölümünden kalan demektir. Örneğin 16'nın 5 ile modunu bulmak için yani 16%5 için: 16 dan 5 tekrar tekrar çıakrılır.

16-5 = 11

11-5 = 6

6-5 = 1

Sonda 1 kaldığı için modu birdir.

Diğer bir öreneğe bakarsak:

16'nın 4 ile modu yani 16%4

16-4 = 12

12-4 = 8

8-4 = 4 (4 ile mod aldığımız için sonuç 4'ten küçük olmalı)

4-4 = 0

Modu 0'dır.

## 7 Atama Operatörleri

Aslında bu bölümde biraz daha karmaşık olan diğer operatörleri göreceğiz.

| Operatör | İşlevi |
|---|---|
| += | Değişkene ekle |
| -= | Değişkenden çıkar |
| *= | Değişkeni çarp |
| /= | Değişkeni böl |
| %= | Değişkeni moduna eşitle |
| **= | Değişkeninin kuvvetini al |
| //= | Değişkeni tam bölümüne eşitle |

Aşağıda örnekleri bulabilirsiniz

[13]:
```
a = 12
a += 5
print("a=",a)

b = 5
b -= 1
print("b=",b)

c = 12
c /= 4
print("c=",c)

d = 2
d *= 5
print("d=",d)

e = 2
e **= 3
print("e=",e)
```

```
a= 17
b= 4
c= 3.0
d= 10
e= 8
```

# 8 Karşılaştırma Operatörleri

| Operatör | İşlevi |
|---|---|
| == | Eşit mi |
| != | Eşit değil mi |
| > | Büyük mü |
| < | Küçük mü |
| >= | Büyük veya eşit mi |
| <= | Küçük veya eşit mi |

- Hatırlarsanız boolean değerlerinden bahsetmiştik. Karşılaştırma ifadeleri bize bu değerleri verir. Dediğim gibi bunları doğru ve yanlış olarak düşünün. Bunların daha detaylı kullanımına ileride değineceğiz.

[14]:
```
a = 12
b = 12
c = 4

print("a b'ye eşit mi:", a==b)
print("a c'ye eşit mi:", a==c)
print("a c'den büyük mü:", a>c)
print("a c'den küçük mü:", a<c)
print("a c'ye eşit  değil mi:" , a!=c)
print("a c'den küçük veya eşit mi:", a<=c)
print("a b'den küçük veya eşit mi:", a<=b)
```

```
a b'ye eşit mi: True
a c'ye eşit mi: False
a c'den büyük mü: True
a c'den küçük mü: False
a c'ye eşit  değil mi: True
a c'den küçük veya eşit mi: False
a b'den küçük veya eşit mi: True
```

# 9 Mantıksal Operatörler

- Operatör listesi:
  - and
  - or
  - not
- kelime anlamı olarak:
  - and = ve
  - or = veya
  - not = değil
  -
- anlamına gelmektedir.

| 1. Değer | **and** | 2. Değer | *Sonuç* |
|---|---|---|---|
| True | **and** | True | =True |
| True | **and** | False | =False |
| False | **and** | False | =False |
| False | **and** | True | =False |

| 1. Değer | **or** | 2. Değer | *Sonuç* |
|---|---|---|---|
| True | **or** | True | =True |
| True | **or** | False | =True |
| False | **or** | False | =False |
| False | **or** | True | =True |

- Örnek kullanım:

deger_1 **and** deger_2 = sonuc

deger_1 **or** deger_2 = sonuc

| **not** operatörü | Değer | *Sonuç* |
|---|---|---|
| **not** | True | =False |
| **not** | False | =True |

- Örnek kullanım
  - not deger_1 = sonuc
  - not deger_2 = sonuc
- "!" ifadesi de yani ünlem işareti de "not" operatörü yerine geçer.
- Örnek kullanım
  - !(deger_1) = sonuc
  - !(deger_2) = sonuc

[15]:
```
print(True or False)
print(False or True)
print(False or False)
print(True and False)
```

```
True
True
False
False
```

[16]:
```
print(not(True))
```

```
False
```

[17]:
```
print(not(False))
```

```
True
```

- Bu ifadeler daha da dallanabilir ve yan yana eklenebilir. Yan yana eklerken işlem önceliği olan parantez içine alınır. Yani diğer bir deyişle parantez içindeki işlemin önceliği vardır.

[18]:
```
print(True and (False or True))
```

```
True
```

- Üstteki ifadeyi parçalarsak şöyle olur:
  1. True and (False or True)
  2. True and True
  3. =True
- Diğer bir örneği ele alalım

[19]:
```
print(False or (True and False))
```

```
False
```

1. False or (True and False)
2. False or False
3. =False

#s If Else blokları (ise değilse blokları)

- Genel kullanımı şu şekildedir
  - if (Koşul 1) :
    - yapılacak işlem veya işlemler
  - else
    - yapılacak işlem veya işlemler
- Her if veya else bloğunun altına, yapılacak işlemler için bir tab tuşu kadar boşluk gerekmektedir. Yani işlemleri yazarken tab tuşuna basıp boşluk bırakmalısınız.
- Eğer if bloğundaki koşul doğru değilse, if bloğundak işlem uygulanmaz. Bilgisayar else bloğuna geçer ve koşula bakmaksızın o işlemleri uygulamaya başlar.
- Yani;

[20]:
```
yasim = 24

if yasim >= 18 :
    print("Ehliyet için uygun.")
else :
    print("Ehliyet için uygun değil.")
```

```
Ehliyet için uygun.
```

[21]:
```
yasim = 15

if yasim >= 18 :
    print("Ehliyet için uygun.")
```

```
else :
    print("Ehliyet için uygun değil.")
```

```
Ehliyet için uygun değil.
```

[22]:
```
isik_durumu = True

if isik_durumu :
    print("Işık açık.")
else :
    print("Işık kapalı.")
```

```
Işık açık.
```

[23]:
```
kapi_durumu = False

if kapi_durumu :
    print("Kapı açık.")
else :
    print("Kapı kapalı.")
```

```
Kapı kapalı.
```

- Bazı durumlarda 1 sayısının True yerine 0 sayısının da False yerine geçebileceğini de öğrenmiştik.

[24]:
```
lamba = 1

if lamba :
    print("Lamba açık.")
else :
    print("Lamba kapalı.")
```

```
Lamba açık.
```

[25]:
```
isitici = 0

if isitici :
    print("Isıtıcı açık.")
else :
    print("Isıtıcı kapalı.")
```

```
Isıtıcı kapalı.
```

## 10 Listeler (List)

Listelerin kısaca formulü aşağıdaki gibidir.

benim_listem = [degisken1, degisken2, .... ]

Yani köşeli parantezler içine elemanları yanı değişkenlerimizi yazarız. Listenin elemanları 0, 1, 2, 3, 4, …. , diye gider. Yani listedeki ilk eleman 0. elemandır.

[26]:
```
benim_listem = ["Ege", "Ada", "Mehmet", "Feride"]
print(benim_listem)
```

```
['Ege', 'Ada', 'Mehmet', 'Feride']
```

[27]:
```
#Listenin ilk elemanı
print(benim_listem[0])
```

```
Ege
```

Listede bir eleman aralığı seçmek istiyorsak da şöyle yaparız

benim_listem[başlangıç : bitiş+1]

örneğin:

[28]:
```
sayilar = [0, 1, 2, 3, 4, 5, 6, 7, 8]
print(sayilar[1: 3])
```

```
[1, 2]
```

Gördüğünüz üzere üç sayısını almadı. Bu sefer 3 ten 6 ya kadar olan sayıları yazdıralım.

[29]:
```
print(sayilar[3:7])
```

```
[3, 4, 5, 6]
```

Diğer bir örnekte ise başlangıçtan 7 ye kadar olan sayıları alalım. Bunun için ":" bu işaretten öncesini boş bırakmamız yeterlidir.

[30]:
```
print(sayilar[:8])
# VEYA
print(sayilar[0:8])
```

```
[0, 1, 2, 3, 4, 5, 6, 7]
[0, 1, 2, 3, 4, 5, 6, 7]
```

Peki ya bir sayıdan itibaren sona kadar olan yeri almak isteseydik?

[31]:
```
print(sayilar[3:])
```

```
[3, 4, 5, 6, 7, 8]
```

Bu seferde ":" bu işaretten önce istediğimiz sayıyı yazıp, işaretten sonrasını boş bırakacaktık. Üstte gördüğümüz gibi.

Aşağıda listeler ile ilgili bazı fonksiyonları göreceğiz.

[32]:
```
#Çantamdaki eşyalar:
print("İlk liste.")
cantamdakiler = ["Kalem","Silgi"]
print(cantamdakiler)
```

```
#Eleman ekleme
print("Defteri ekledik.")
cantamdakiler.append("Defter")
print(cantamdakiler)

#Eleman çıkarma Yöntem 1
print("Kalemi listeden çıkardık.")
cantamdakiler.remove("Kalem")
print(cantamdakiler)

#Eleman çıkarma Yöntem 2
print("Listedeki ilk elemanı çıkardık.")
cantamdakiler.pop(0)
print(cantamdakiler)
```

```
İlk liste.
['Kalem', 'Silgi']
Defteri ekledik.
['Kalem', 'Silgi', 'Defter']
Kalemi listeden çıkardık.
['Silgi', 'Defter']
Listedeki ilk elemanı çıkardık.
['Defter']
```

[33]:
```
# Yeni bir listeyle devam edelim:
evimdeki_esyalar = ["Yatak","Kanepe","Televizyon"]
print(evimdeki_esyalar)

# Yatak ile kanepenin arasına yeni eşya ekleyelim
evimdeki_esyalar.insert(1, "Fırın")
print(evimdeki_esyalar)

# Listenin uzunluğunu bulmak için
evimdeki_esya_sayisi = len(evimdeki_esyalar)
print("Evimdeki eşya sayısı:")
print(evimdeki_esya_sayisi)
```

```
['Yatak', 'Kanepe', 'Televizyon']
['Yatak', 'Fırın', 'Kanepe', 'Televizyon']
Evimdeki eşya sayısı:
4
```

Listelerle bazı işlemler yapalım

[34]:
```
daginik_liste = [2, 6, 3, 7, 4, 9, 5]

print("Dağınık liste:",daginik_liste)
# Üstteki listeyi başka bir değişkene kopyalıp sıralayalım
```

```
a = daginik_liste
a.sort()
print("Sıralanmış liste:",a)

# Dagınık listeyi tersten sıralayalım
b = daginik_liste
b.sort()
b.reverse()
print("Tersten sıralanmış liste:",b)
```

```
Dağınık liste: [2, 6, 3, 7, 4, 9, 5]
Sıralanmış liste: [2, 3, 4, 5, 6, 7, 9]
Tersten sıralanmış liste: [9, 7, 6, 5, 4, 3, 2]
```

[35]:
```
# Yeni yöntemler keşfedelim
meyve_sepeti = ["Elma", "Elma", "Elma", "Armut", "Armut", "Portakal"]
print("Meyve sepetim:",meyve_sepeti)

#  Sepetimde kaç elma var
print("Elma sayısı:",meyve_sepeti.count("Elma"))
```

```
Meyve sepetim: ['Elma', 'Elma', 'Elma', 'Armut', 'Armut', 'Portakal']
Elma sayısı: 3
```