



KONU : PYTHON PROGRAMLAMA DİLİNE GİRİŞ

PYTHON PROGRAMLAMA DİLİ İLE İLGİLİ TEMEL BİLGİLER

❑ Python; nesne yönelimli, yorumlanabilen, birimsel (modüler) ve etkileşimli bir programlama dilidir.

❑ Python, Guido Van Rossum adlı Hollandalı bir programcı tarafından 90'lı yılların başında geliştirilmeye başlanmış bir programlama dilidir. Zannedildiğinin aksine bu programlama dilinin adı piton yılanından gelmez... Guido Van Rossum bu programlama dilini, “*The Monty Python*” adlı bir İngiliz komedi grubunun, “*Monty Python’s Flying Circus*” adlı gösterisinden esinlenerek adlandırmıştır.



PYTHON PROGRAMLAMA DİLİNİN ÖZELLİKLERİ

- Nesneye yönelik bir programlama dilidir.
- Özgürdür.
- Derlenebilir
- Taşınabilirdir.
- Güçlü ve hızlıdır.
- Yazılımı kolay ve sadedir.
- Ticari uygulamalar geliştirilebilir.



PYTHON PROGRAMLAMA DİLİ NERELEERDE KULLANILIR

- Veritabanı Programcılığı
- GUI (Graphic User Interface) Programcılığı
- İnternet Programcılığı



PYTHON DİLBİLGİSİ

Diğer programlama dillerinde olduğu gibi Python'da da kod yazımında uyulması gereken kurallar vardır. Bir dili kullanabilmek için bu kuralları iyi bilmek ve uygulamak gereklidir.

- Python büyük-küçük harf ayrımı yapar.
- Komutların sonlarında ; (noktalı virgül) ve benzeri işleçler konulmaz.
- Pythonda kod yazarken girintiler kullanmak zorundasınız.
- Pythonda değişkenleri tanımlama yoktur.

Blok1

Blok1'e ait kodlar

Blok1 başlangıcından daha içeriden başlar.

Eğer bu blok içerisinde de başka bloklar varsa

O bloğa ait kodlar da

O bloktan daha içeride başlar.

Tekrar bir önceki girintiye döndüğümüzde yine BLOK1'e ait kodları

Blok2

Blok1'le aynı seviyeye geldiğimizde yeni bir blok başlamış olur.

PYTHON OPERATÖRLERİ

❖ Aritmetik Operatörler

Operatör	İşlem	Örnek
=	Atama Operatörü	A=4, b='Oku'
+	Toplama	A=4+86
-	Çıkarma	A=86-52
*	Çarpma	A=9*86
/	Bölme	A=86/12
**	Kuvvet alma	A=8**3 (8'in 3.kuvveti)

PYTHON OPERATÖRLERİ

❖ Mantıksal Operatörler

Operatör	İşlem	Örnek
and	Ve işlemi	<code>a == 3 and b == 12</code>
or	Veya İşlemi	<code>a == 5 or b == 43</code>
not	Değil İşlemi	<code>not a % 2 == 0</code>

PYTHON OPERATÖRLERİ

❖ Karşılaştırma Operatörleri

Operatör	İşlem	Örnek
>	Büüklük	A>B (A, B' den büyüktür)
>=	Büüklük ya da eşitlik	A>=B (A, B' ye eşit veya büyük)
<	Küçüklük	A<B (A, B' den küçüktür)
<=	Küçüklük ya da eşitlik	A<=B (A, B' den küçük veya eşit)
==	Eşitlik	A=B (A, B' ye eşit)
!=	Eşit değil	A<>B (A, B' ye eşit değil)

PYTHON PROGRAMLAMA DİLİNDE DEĞİŞKENLER



Uyarı !

- Diğer dillerde de olduğu gibi bir değişken RAKAM ile başlayamaz. İlk karakter bir harf veya _ (altçizgi) olmak zorundadır.
- Harf, Rakam ve _ (alt çizgi) haricinde bir karakter içeremez (örn: \$, #, *, ? veya boşluk gibi).
- Aksi belirtilmedikçe tüm değişkenler yerel olarak algılanırlar.

PYTHON DİLİNDE DEĞİŞKENLER ve VERİ TİPLERİ

❖ Sayısal Değişkenler

Fonksiyon	Anlamı
int(n)	n sayısını tamsayıya (integer) dönüştürür. Ör: int(010)
float(n)	n sayısını kayan noktalı sayıya (float) dönüştürür. Ör: float(13)
long(n)	n sayısını uzun tamsayıya (long integer) dönüştürür. Ör: long(13)
oct(n)	n sayısının sekizlik tabandaki karşılığını verir. Ör: oct(13)
hex(n)	n sayısının onaltılık tabandaki karşılığını verir. Ör: hex(13)

PYTHON DİLİNDE DEĞİŞKENLER ve VERİ TİPLERİ

❖ String Değişkenler

String bir değişkene değer atamak için “ (çift tırnak) veya ‘ (tek tırnak) ifadesi kullanılır. Eğer karakter dizisi belirtilirken çift tırnak kullanılırsa o karakter dizisi içerisinde çeşitli özel karakterler (%s , %d vs.) aranır , varsa değiştirilir.

```
>>>ad=“alimurat”
```

```
>>>ad=‘alimurat’
```

```
>>>a=9
```

```
>>>b=“5”
```

```
>>>c=a+b
```

Traceback (most recent call last):

File "<pyshell#11>", line 1, in <module>

a+b

TypeError: unsupported operand type(s) for +: 'int' and 'str'

PYTHON DİLİNDE DEĞİŞKENLER ve VERİ TİPLERİ

❖ Listeler

Bir liste birden çok string yada sayı sabitini belirli bir sırada barındıran değişkenlerden veya sabitlerden oluşur ve oluştururken [] (köşeli parantez) ifadesi kullanılır. Liste içerisindeki elemanların indeks numarası 0 (sıfır) ile başlar. Listenin elemanlarına ulaşmak için *liste[indeks numarası]* şeklinde bir yazım kullanılırız. Dikkat edilmesi gereken diğer nokta ise indeks değerinin mutlaka bir tam sayı olması zorunluluğudur.

Örnek :

```
>>> adi=['ali', 'murat', 'garipcan']
>>> adi
>>> ['ali', 'murat', 'garipcan']

>>> adi[0]
>>> ali

>>> adi[-3]
>>> 'ali'

>>> adi[1:2]
>>> 'murat'
```

PYTHON DİLİNDE DEĞİŞKENLER ve VERİ TİPLERİ

❖ En Çok Kullanılan Liste Fonksiyonları

- **count ()**: Listede bir elemanın kaç defa tekrarlandığını verir
`programlar.count('python')`
- **extend ()**: İki listeyi toplar.
`programlar.extend(['Java'])`
- **index()**: İstenilen bir elemanın liste içindeki indeksini verir.
`programlar.index('istenen değer')`
- **pop()**: Listenin son elemanını çıkartır.
`programlar.pop()`
- **remove ()**: Herhangi bir elemanı listeden çıkartmak için kullanılır.
`Programlar.remove('Java')`
- **reverse()**: Listeyi tersten yazdırır.
- **sort()**: Liste elemanlarını sıralamak için kullanılır.
- **len()**: Listenin kaç elemandan oluştuğunu bulur.
`len(programlar)`

PYTHON DİLİNDE DEĞİŞKENLER ve VERİ TİPLERİ

❖ Tüpler:

Tüpler de listeler gibidir fakat bir tüpün içeriğini değiştiremezsiniz. Listelerden farklı olarak tanımlama işlemi yaparken parantezler kullanılır.

```
>>> meyvalar=('Elma', 'Armut', 'Portakal')
```

```
>>> meyvalar[-2]
```

```
>>> 'Portakal'
```

```
>>> (a,b,c)=(100, 'Murat', '000,1')
```

```
>>> a
```

```
>>> 100
```

```
>>> a*c
```

```
>>> 0,1
```

KONU : OPERATÖRLER, ŞARTLI İFADELER, DÖNGÜLER

OPERATÖRLER

OPERATÖR	ANLAMI	ÖRNEK KULLANIMI
=	Atama Operatörü	A=4, b='Oku'
+	Toplama	A=4+86
-	Çıkarma	A=86-52
*	Çarpma	A=9*86
/	Bölme	A=86/12
**	Kuvvet alma	A=8**3 (8'in 3.kuvveti)
<	Küçük	if x<y
<=	Küçük veya Eşit	if x<=y
>	Büyük	if x>y
>=	Büyük veya Eşit	if x>=y
==	Eşit	if x==y
!=	Eşit Değil	if x!=y

OPERATÖRLER

OPERATÖR	ANLAMI	ÖRNEK KULLANIMI
%	Kalanı verir	5%2 Sonuç:1
//	Bölümü verir	20//2 Sonuç=10
Pow(x,y)	Kuvvet alır	Pow(3,4) Sonuç=81
abs(x)	Mutlak Değer	Abs(-10) Sonuç=10
Or	Veya (Biri Doğru Olunca Sonuç Doğru)	X or Y
And	Ve(Sadece İkisi doğru olunca sonuç doğru)	X and Y
Not	Değil(Doğru ise yanlış, Yanlış ise doğru yapar)	Not X

OPERATÖRLER

Örnekler:

```
>>> x=20;y=5 #x ve y değişkenlerine değer atanması
>>> x+y #toplama işlemi
25
>>> x-y #çıkarma işlemi
15
>>> x/y #bölme işlemi
4.0
>>> x*y #çarpma işlemi
100
>>> x**y #kuvvet alma
3200000
>>> pow(x,y) #kuvvet alma
3200000
>>> x%y #kalanı verir
0
>>> x//y #bölümü verir
4
>>> abs(y-x) #mutlak değer işlemi
15
>>>
```

OPERATÖRLER

Örnekler:

```
>>> x=3 ; y=2   değerleri için ;
```

Eşittir Operatörünün Kullanımı;

```
>>> if x==y:                # == eşittir operatörünün kullanımı
    print('x y değerine eşit')
else: print('x y değerine eşit değil')
```

```
x y değerine eşit değil
```

Eşit Değildir Operatörünün Kullanımı;

```
>>> if x!=y:                # != eşit değildir operatörünün kullanımı
    print('x y değerine eşit değil')
else: print('x y değerine eşit')
```

```
x y değerine eşit değil
```

OPERATÖRLER

Örnekler:

```
>>> x=3;y=2
```

Büyüktür Operatörünün Kullanımı;

```
>>> if x>y:                # > büyüktür operatörünün kullanımı
    print('x y değerinden büyüktür')
else: print('x y değerine eşit veya küçüktür')
```

```
x y değerinden büyüktür
```

Or (veya) Operatörünün Kullanımı;

```
>>> x=2;y=2
>>> if x>y or x<y:        # or veya operatörünün kullanımı
    print('x y değerine eşit değildir')
else: print('x y değerine eşittir')
```

```
x y değerine eşittir
```

ŞARTLI İFADELER

İF ELSE

if <şart>:

doğruysa işletilecek blok

elif<şart>:

doğruysa işletilecek blok

else: şartlar yanlışsa işletilecek blok

ŞARTLI İFADELER

IF ELSE

Örnekler: Sayının çift veya tek olduğunu bulan program

Sayı Tek Olduğunda

```
>>> x=15
>>> if x%2==0:
    print(x,' çift sayıdır.')
else:print(x,' tek sayıdır')

15  tek sayıdır
```

Sayı Çift olduğunda

```
>>> x=10
>>> if x%2==0:
    print(x,' çift sayıdır.')
else:print(x,' tek sayıdır')

10  çift sayıdır.
```

IF ELSE

```
>>> z='b'
>>> if 'x'<z:
        print('Merhaba')
        print('Ahmet')
```

```
>>> z='y'
>>> if 'x'<z:
        print('Merhaba')
        print('Ahmet')
```

```
Merhaba
Ahmet
```

Birinci ifadede z değişkenine atanan b değeri x değerinden büyük olduğu için şart sağlanamamıştır. Ancak else satırı olmadığı için ekrana herhangi bir şey yazılmamıştır.

İkinci ifadede z değişkenine atanan y değeri x değerinden büyük olduğu için şart sağlanmıştır ve ekrana print komut satırındaki ifadeler yazılmıştır.

Örnekler:

```
x=15
y=5
z=11
print (x,y,z)
if x<y:
    if x<z: print ('x en küçük')
    else:print ('z en küçük')
else: print ('y en küçük')
```

İç içe if else yapısı kullanılarak sayılar arasında en küçük olanın bulunması sağlanmıştır.

Programı çalıştırdığımızda sonuç aşağıdaki gibi olur.

```
>>> ===== RESTART =====
>>>
15 5 11
y en küçük
>>>
```

İTERASYON (TEKRARLANMA-DÖNGÜ) FOR DÖNGÜSÜ

for i in range (başlangıç,bitiş,artışmiktari):

**bitiş sayısına veya daha büyük bir
sayıya ulaşana kadar işletilecek blok**

**Bitiş sayısına veya daha büyük bir sayıya
ulaşıldıktan sonra işletilecek blok**

İTERASYON (TEKRARLANMA-DÖNGÜ) FOR DÖNGÜSÜ

Örnekler:1 den 10'a kadar sayıların toplamını bulan program.

```
x=11;toplām=0
for i in range(1,x):
    toplām=toplām+i
print (toplām)
```

Programın Sonucu:

```
>>> ===== RESTART ===
```

```
>>>
```

```
55
```

İTERASYON (TEKRARLANMA-DÖNGÜ) FOR DÖNGÜSÜ

Örnekler:1 den 10'a kadar tek sayıların toplamını bulan program.

```
x=10; toplam=0
for i in range(1,x,2):
    toplam=toplam+i
print(toplam)
```

Programın Sonucu:

```
>>> ===== RESTART =====
```

```
>>>
```

```
25
```

İTERASYON (TEKRARLANMA-DÖNGÜ) FOR DÖNGÜSÜ

Örnekler: Klavyeden girilen sayının kendisinden küçük tam bölenlerini bulan program.

```
n=int(input('Sayıyı giriniz:'))
print(n,'sayısının tam bölenleri')
for i in range(1,n):
    if n%i==0:
        print(i)
```

Programın Sonucu:

```
>>> ===== RESTART =====
>>> Sayıyı giriniz:20
20 sayısının tam bölenleri
1
2
4
5
10
>>>
```

İTERASYON (TEKRARLANMA-DÖNGÜ) WHILE DÖNGÜSÜ

while <şart>:

Şart Doğru olduğu sürece işletilecek blok

Şart sağlanmadığında devam edilecek blok

- While ifadesi doğru olduğu müddetçe devam eden döngüdür.
- While döngüsü ile diğer döngü ve şart ifadeleri beraber içiçe kullanılabilir.

İTERASYON (TEKRARLANMA-DÖNGÜ)

WHILE DÖNGÜSÜ

Örnekler:

```
y=0
x=3
iterasyonsayisi=x
while(iterasyonsayisi>0):
    y=y+x
    print('y=',y,'iterasyonsayisi=',iterasyonsayisi)
    iterasyonsayisi=iterasyonsayisi-1
```

iterasyonsayisi değeri 0'dan büyük iken y değişkeninin değeri her defasında x değeri kadar arttıran program.

Program çalıştırıldığında sonuç aşağıdaki gibi olur.

```
>>> ===== RESTART ===>
>>>
y= 3 iterasyonsayisi= 3
y= 6 iterasyonsayisi= 2
y= 9 iterasyonsayisi= 1
```

WHILE DÖNGÜSÜ

Örnekler:

```
x=10
i=1
print(x, 'sayısını tam bölen sayılar')
while(i<=x):
    if x%i==0:
        print(i)
    i=i+1
```

Yukarıdaki program x değerini tam bölen sayıların bulunmasını sağlar. Program çalıştırıldığında sonuç aşağıdaki gibi olur.

```
10 sayısını tam bölen sayılar
1
2
5
10
>>>
```

İTERASYON (TEKRARLANMA-DÖNGÜ)

WHILE DÖNGÜSÜ

Örnekler: Girilen Sayının Faktöriyelini while döngüsü ile bulan program

```
n=int(input('Faktoriyeli alınacak sayıyı giriniz:'))
fakt=1
while(n>0):
    fakt=fakt*n
    n=n-1
print('Sonuç=',fakt)
```

Program çalıştırıldığında sonuç aşağıdaki gibi olur.

```
>>> ===== RESTART =====
>>>
Faktoriyeli alınacak sayıyı giriniz:5
Sonuç= 120
>>>
```

KAYNAKLAR

- MIT Üniversitesinin ders notları

<http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-00-introduction-to-computer-science-and-programming-fall-2008/lecture-videos/>

- <http://www.python.org/>