



ANKARA UNIVERSITY CYBER CLUB

Python3 Giriş



~ Python Hakkında

**Yorumlanan bir dildir.*

**Nesne yönelimli, imperative ve fonksiyonel programlama*

**Dinamik tip sistemi*

**Otomatik hafıza yönetimi*

**Kod blokları girintiler ile birbirinden ayrılır.*

~ Neden Python

- *Özgür. Kullanmakta özgürsünüz, bir şirkete bağlı olmak zorunda değilsiniz. Üstelik ücretsiz.*
- *Yapısı sade. Okuması ve yazması çok kolay. Çok hızlı öğrenilebiliyor.*
- *“Dinamik” dil. Yorumlayıcıyla çalışıyor. Çok karmaşık işlemleri basit komutlarla yaptırabilirsiniz.*
- *Etkileşimli. Yorumlayıcı penceresinde ard arda komutlar verip işlemler yapabilirsiniz.*
- *Aklınıza gelen bir fikri çabucak bir program oluşturarak deneyebilirsiniz.*
- *OOP’yi destekler ama mecbur tutmaz. Basit işler için class tanımlamak zorunda değilsiniz.*
- *Genel kullanım alanı geniş ve yazılım sanayisinde çok seviliyor. Bu sayede sürekli geliştiriliyor. Öğretici kaynak bulmak kolay.*
- *Bilimsel araştırmalarda çok yaygın olarak kullanılıyor. Çeşitli bilim dallarında kullanılmak üzere özel hazırlanmış kütüphaneleri var. Bu kütüphaneler kendini bu işe adanmış profesyonel yazılım ekipleri tarafından hazırlanıyor, yoğun şekilde test ediliyor, ve yine özgür.*


~ Neden Python

```
#include <iostream>

using namespace std;

int main()
{
    int a=3;
    int b=2;
    cout<<a+b;
}
```

```
V V V 3+2
5
V V V
```





~ Python Kurulum

GNU/Linux :

**Neredeyse bütün GNU/Linux dağıtımlarında Python programlama dili kurulu olarak gelmektedir*



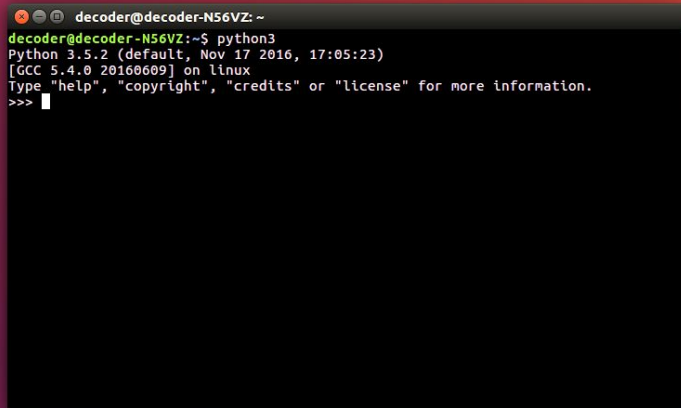
~ Python Kurulum

Windows :

Windows sürümlerinin hiçbirinde Python kurulu değildir, windows kullanıcıları, Python'ı sitesinden indirip kurması gerekmektedir. `http://www.python.org/downloads` adresinden indirilen .exe uzantılı dosya yardımı ile Python kurulabilmektedir.

~ Etkileşimli Kabuk(interactive shell)

Etkileşimli kabuk görüntüsü , buradan çıkmak için ctrl+d , ctrl+z , ve her sistemde geçen `import sys; sys.exit()`

A terminal window with a black background and white text. The title bar shows 'decoder@decoder-N56VZ: ~'. The text inside the terminal shows the user has run 'python3', resulting in the Python 3.5.2 prompt '>>>'. The prompt is followed by a white cursor character. The background of the slide is a red-to-purple gradient.

```
decoder@decoder-N56VZ: ~  
decoder@decoder-N56VZ:~$ python3  
Python 3.5.2 (default, Nov 17 2016, 17:05:23)  
[GCC 5.4.0 20160609] on linux  
Type "help", "copyright", "credits" or "license" for more information.  
>>> █
```

~ Karakter Dizilerine Giriş

“Ankara Üniversitesi”

type(“AUCC”) = <class ‘str’>

“ankara”+“üniversitesi” = ankaraüniversitesi

*>>> “w”*3 + “.ankara” + “.edu” + “.tr”*

~ Sayılara Giris (aritmetik islemler)

```
>>>5
```

```
>>>5.5
```

```
>>>5-2
```

```
>>>10*10
```

```
>>>12/5
```

```
>>>5+5
```

```
>>>5%2
```

```
>>>2**3
```

~ Degiskenler

Kurallar:

Değişken adları bir sayı ile başlayamaz.

```
1Test = 12
```

```
test = 12
```

Değişken adları aritmetik işlemlerle başlayamaz.

Değişken adları ya bir alfabe harfiyle ya da "_" işaretiyle başlamalıdır.

Değişken adları içinde Türkçe karakterler kullanılabilir.

~ Degiskenler

Değişken adları içinde boşluk yerine "_" kullanılabilir.

Test degisken = 12

test_degisken = 12

Yasaklı kelimeler : 'False', 'None', 'True', 'and', 'as', 'assert', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield' bu kelimelerin hiç biri değişken olarak kullanılamaz.

>>>import keyword >>> keyword.kwlist



~ Degiskenler

elif = “ tatlı kız” izin vermeyecektir

peki kaç tane yasaklı kelime var hemen öğrenmek istiyoruz diyelim ne yapabiliriz?

type gibi olan len() komutunu kullanarak len(keyword.kwlist) komutu ile bastırılan kelimelerin sayısını bulabiliriz.

Uygulama = dairenin alanı hesaplayan kodu yazalım.

~ Dönüştürme İşlemleri

int()

str()

float()

complex()

Peki type(1234) ve type("1234") arasındaki fark ne olabilir?

~ Dönüştürme İşlemleri

```
>>> a = 12
>>> b = str(a)
>>> type(b)
<class 'str'>
>>> c = int(b)
>>> type(c)
<class 'int'>
>>> float(c)
12.0
```



~ **print()** fonksiyonu

Python2 ile Python3 arasındaki farklardan biri print komutu/fonksiyondur.

python2:

```
>>> print "AUCC"
```

python3:

```
>>> print("AUCC")
```

~ print() fonksiyonu

```
>>> print ("https://", "www.aucyberclub.", "org/")
https:// www.aucyberclub. org/
>>> print ("https://", "www.aucyberclub.", "org/", sep="")
https://www.aucyberclub.org/
>>> █
```


~ input() fonksiyonu

kullanıcıdan bilgi almak için kullanılan fonksiyondur.

```
>>>değişken = input("değişkeni giriniz")
```

```
>>> isim = input("lütfen isminizi giriniz:\n")
lütfen isminizi giriniz:
AUCC
>>> print("girilen isim ",isim, ".")
girilen isim  AUCC .
>>>
```



~ **input()**

input() fonksiyonuyla kullanıcıdan alınan değer her zaman karakter dizisidir.

Bu sebeple aritmetik işlemler yaparken tip dönüştürme işlemlerinden yararlanılır.

~Kosul ifadeleri

Python'da koşul ifadeleri üç anahtar kelimeyle tanımlanır

if

else

elif

~ Kosul Ifadeleri

```
a = int(input("bir sayı giriniz\n"))

if (a <=10):
    print ("girilen sayı 10'dan küçük")
elif (a>10 and a<15):
    print ("sayı 10 ile 15 arasında")
else:
    print ("sayı 15'ten büyüktür")
```

~ While Döngüsü

While koşul ifadesi :

```
decoder@decoder-N56VZ: ~  
IPython 2.4.1 -- An enhanced Interactive Python.  
?      -> Introduction and overview of IPython's features.  
%quickref -> Quick reference.  
help    -> Python's own help system.  
object?  -> Details about 'object', use 'object??' for extra details.  
  
In [1]: a=1  
  
In [2]: while a<10:  
...:     a+=1  
...:     print("N00b Alert!!")  
...:  
N00b Alert!!  
N00b Alert!!  
N00b Alert!!  
N00b Alert!!  
N00b Alert!!  
N00b Alert!!  
N00b Alert!!  
N00b Alert!!  
N00b Alert!!  
N00b Alert!!
```

~ For Döngüsü

```
decoder@decoder-N56VZ: ~  
  
In [3]: sayilar="05364183037"  
  
In [4]: for i in sayilar:  
...:     if int(i) > 3:  
...:         print(i)  
...:  
5  
6  
4  
8  
7  
  
In [5]:
```

~ Range() Fonksiyonu

Range fonksiyonu bize bir liste oluşturur.

3 farklı kullanım yöntemi vardır.

```
#for i range(1,10):
```

```
    print(i)
```

aynı şekilde range(5,10) range(1,10,2) kullanımları vardır.

~ Range() Fonksiyonu

```
>>> for i in range(10,0,-1):  
...     print(i)  
...  
10  
9  
8  
7  
6  
5  
4  
3  
2  
1
```




Parola dogrulama :

```
parola.py (-/Desktop) - gedit
Open  Save
while True:
    parola = input("parola giriniz: ")
    if not parola:
        print("!!!!")
    elif len(parola) in range(8,16):
        print("Yeni parolaniz", parola)
        break
    else:
        print("parola 8 karakterden uzun 16 karakterden kısa olmalı")
Python Tab Width: 8 Ln 14, Col 1 INS
```

~ break

```
decoder@decoder-N56VZ: ~  
  
In [2]: while True:  
...:     parola=input("parola pls")  
...:     if len(parola)<5:  
...:         print("parola 5 den az olmamali")  
...:     else:  
...:         print("parola belirlendi")  
...:         break  
...:   
parola pls
```

~ continue

```
decoder@decoder-N56VZ:~/Desktop$ ipython3
Python 3.5.2 (default, Nov 17 2016, 17:05:23)
Type "copyright", "credits" or "license" for more information.

IPython 2.4.1 -- An enhanced Interactive Python.
?      -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help    -> Python's own help system.
object? -> Details about 'object', use 'object??' for extra details.

In [1]: while True:
...:     s=input("Bir sayi girin:")
...:     if s=="iptal":
...:         break
...:     if len(s) <=3:
...:         continue
...:     print("En fazla üç haneli")
...:
Bir sayi girin:1234
En fazla üç haneli
Bir sayi girin:123
Bir sayi girin:█
```

~ pass

```
decoder@decoder-N56VZ: ~  
  
In [4]: while True:  
...:     sayi=int(input("sayi girin"))  
...:     if sayi==0:  
...:         break  
...:     elif sayi <0:  
...:         pass  
...:     else:  
...:         print(sayi)  
...:  
sayi girin-1  
sayi girin
```

~ Listeler, Demetler, Sözlükler

Python'da kullanılan diğer veri tipleri liste(list), demet(tuple) ve sözlük(dictionary)tür.

Bu üç veri tipini tanımlarken birbirinden nasıl ayırıyoruz?

değişken_liste = []

değişken_demet = ()

değişken_sözlük = { }

~ Liste

Birden fazla değişkeni içinde barındıran veri tipidir.

Python değişik tiplerdeki değişkenlerden oluşan liste yapısını destekler

```
>>> liste = ["AUCC",1,3,4.0]
>>> print (liste)
['AUCC', 1, 3, 4.0]
>>> type(liste)
<class 'list'>
>>>
>>>
>>>
```

~ Liste

list() fonksiyonu da liste oluşturmayı, ayrıca diğer veri tiplerinden listeyi geçisi sağlar.

```
>>> string = "AUCC"
>>>
>>> type(string)
<class 'str'>
>>>
>>> liste = list(string)
>>>
>>> liste
['A', 'U', 'C', 'C']
>>>
>>> type(liste)
<class 'list'>
```

~ Liste

Listedeki elemanlara nasıl ulaşırız?

```
>>> liste = ["elma","kiraz",1,2,3]
>>> len(liste)
5
>>> liste[0]
'elma'
>>> liste[3]
2
>>> for eleman in liste:
...     print (eleman)
...
elma
kiraz
1
2
3
```


~ Liste

Listedeki elemanlara nasıl ulaşırız?

```
>>> liste = ["elma","armut","kiraz",1,2,3,4,5]
>>> dilim1 = liste[0:4]
>>> dilim1
['elma', 'armut', 'kiraz', 1]
>>> dilim2 = liste[2:4]
>>> dilim2
['kiraz', 1]
```

```
>>> dilim3 = liste[::2]
>>> dilim3
['elma', 'kiraz', 2, 4]
>>> dilim4 = liste[:5:3]
>>> dilim4
['elma', 1]
>>>
```

~ Liste

Listeye eleman ekleme

```
>>> liste = ["elma","kiraz",1,2,3]
>>> liste+["armut"]
['elma', 'kiraz', 1, 2, 3, 'armut']
>>> liste+[12]
['elma', 'kiraz', 1, 2, 3, 12]
>>>
>>>
>>> liste = liste+[12]
>>> liste
['elma', 'kiraz', 1, 2, 3, 12]
>>> liste = liste+["armut"]
>>> liste
['elma', 'kiraz', 1, 2, 3, 12, 'armut']
>>>
>>> █
```

~Liste

Listelerin eşitliği

```
>>> liste1 = ["elma","kiraz",1,2,3]
>>> liste2 = liste1
>>> liste1
['elma', 'kiraz', 1, 2, 3]
>>> liste2
['elma', 'kiraz', 1, 2, 3]
>>> liste1[0] = 8
>>> liste1
[8, 'kiraz', 1, 2, 3]
>>> liste2
[8, 'kiraz', 1, 2, 3]
>>>
```

~ Liste

list() fonksiyonuyla listelerin eşitliği

```
>>> liste1 = ["elma","kiraz",1,2,3]
>>> liste2 = list(liste1)
>>> liste1
['elma', 'kiraz', 1, 2, 3]
>>> liste2
['elma', 'kiraz', 1, 2, 3]
>>> liste1[0]=8
>>> liste1
[8, 'kiraz', 1, 2, 3]
>>> liste2
['elma', 'kiraz', 1, 2, 3]
>>>
```

~ Liste Metotlari

- *append()*
- *extend()*
- *insert()*
- *remove()*
- *pop()*
- *sort()*
- *index()*
- *count()*
- *copy()*
- *clear()*

~ Liste Metotları

```
>>> liste = ["elma","kiraz",1,2,3,"elma",2,2]
>>> liste.append("5")
>>> liste
['elma', 'kiraz', 1, 2, 3, 'elma', 2, 2, '5']
>>> yeni_liste = [1,2,3]
>>> liste.extend(yeni_liste)
>>> liste
['elma', 'kiraz', 1, 2, 3, 'elma', 2, 2, '5', 1, 2, 3]
>>> liste.insert(2,"çilek")
>>> liste
['elma', 'kiraz', 'çilek', 1, 2, 3, 'elma', 2, 2, '5', 1, 2, 3]
>>> liste.remove("kiraz")
>>> liste
['elma', 'çilek', 1, 2, 3, 'elma', 2, 2, '5', 1, 2, 3]
>>> liste.remove(2)
>>> liste
['elma', 'çilek', 1, 3, 'elma', 2, 2, '5', 1, 2, 3]
```

~ Liste Metotları

```
>>> liste
['elma', 'çilek', 1, 3, 'elma', 2, 2, '5', 1, 2]
>>> liste.pop()
2
>>> liste
['elma', 'çilek', 1, 3, 'elma', 2, 2, '5', 1]
>>> liste.index("çilek")
1
>>> liste.index(2)
5
>>> liste.count(2)
2
>>> liste_2 = liste.copy()
>>> liste_2
['elma', 'çilek', 1, 3, 'elma', 2, 2, '5', 1]
>>> liste_2.clear()
>>> liste_2
[]
>>> █
```

~ Demet

- *Demetler de listeler gibi birden fazla değişkeni içinde barındırır.*
- *Listeler gibi farklı veri tiplerinde değişkenleri barındırabilir.*
- *Demet tanımlamak için birden fazla yol vardır.*

~ Demet

Demet tanımlama yöntemleri

```
>>> demet = ("AUCC",2017, "python dersleri")
>>> type(demet)
<class 'tuple'>
>>> demet2 = "AUCC",2017,"python dersleri"
>>> type(demet2)
<class 'tuple'>
>>> demet3 = tuple("AUCC")
>>> demet3
('A', 'U', 'C', 'C')
>>> type(demet3)
<class 'tuple'>
>>> █
```

~ Demet

Demetlerle listeler arasındaki fark nedir?

- *Listeler sonradan değiştirilmeye elverişlidir. Demetler değiştirme işlemlerine elverişli değildir.*
- *Demetler döngülerde listelerden daha hızlı sonuç verir.*

~demet

Demet elemanlarına erişim listelerdeki gibidir.

```
>>> demet
('AUCC', 2017, 'python dersleri')
>>> demet[0]
'AUCC'
>>> demet[-1]
'python dersleri'
>>>
>>>
```

~Demet Metotlari

- *index()*
- *count()*

```
>>> demet =tuple("Ankara Üniversitesi")
>>> demet
('A', 'n', 'k', 'a', 'r', 'a', ' ', 'Ü', 'n', 'i', 'v', 'e', 'r', 's', 'i', 't', 'e', 's', 'i')
>>> demet.index("Ü")
7
>>> demet.count("a")
2
>>>
```

~Sözlük

- *Python'da sözlük veri tipi iki değişken arasında bağıntı kurulmasını sağlar.*
- *Bu değişkenlerden ilki "key", ikincisi "value" olarak adlandırılır.*
- *Sözlüklerde her "key" değeri bir "value" ile eşleşir.*
- *Sözlükleri tanımlamak için :*

```
>>>sözlük = {anahtar1 : değer1 , anahtar2 : değer2 }
```

~Sözlük

```
>>> sözlük = {"a":0,  
...          "b":1,  
...          "c":3,  
...          "d":4,  
...          "e":5,  
>>> sözlük  
{'d': 4, 'a': 0, 'e': 5, 'c': 3, 'b': 1}  
>>>  
>>>  
>>> sözlük.keys()  
dict_keys(['d', 'a', 'e', 'c', 'b'])  
>>> sözlük.values()  
dict_values([4, 0, 5, 3, 1])  
>>>  
>>> sözlük["b"]  
1
```

~Sözlük

- *Sözlüklerde verilere erişirken listelerdeki gibi sıra numarasıyla erişemeyiz. Çünkü sözlüklere eklenen veriler karışık sıralanır.*

```
>>> sözlük["c"]
3
>>> sözlük.items()
dict_items([('d', 4), ('a', 0), ('e', 5), ('c', 3), ('b', 1)])
>>> sözlük[0]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 0
>>>
```

~Sözlük

- Sözlüklerde key değerleri değiştirilemez veri tipleri olmak zorundadır.
- Ancak value değerleri için bu durum geçerli değildir.

```
>>> sözlük["f"]=6
>>> sözlük.items()
dict_items([('d', 4), ('e', 5), ('c', 3), ('b', 1), ('a', 0), ('f', 6)])
>>> sözlük["g"]=7
>>> sözlük.items()
dict_items([('d', 4), ('e', 5), ('g', 7), ('c', 3), ('b', 1), ('a', 0), ('f', 6)])
>>>
>>> sözlük["h"]=[1,2,3]
>>> sözlük.items()
dict_items([('d', 4), ('e', 5), ('g', 7), ('c', 3), ('b', 1), ('a', 0), ('h', [1, 2, 3]), ('f', 6)])
```


~ Hata Ayıklama(Try ,expect)

```
 -*- coding: utf-8 -*-  
  
ilk_sayi = input("ilk sayı: ")  
ikinci_sayi = input("ikinci sayı: ")  
  
try:  
    sayi1 = int(ilk_sayi)  
    sayi2 = int(ikinci_sayi)  
    print(sayi1, "/", sayi2, "=", sayi1 / sayi2)  
except (ZeroDivisionError, ValueError):  
    print("hata oluştu")
```

~try, except, as

```
# -*- coding: utf-8 -*-

ilk_sayı = input("ilk sayı: ")
ikinci_sayı = input("ikinci sayı: ")

try:
    sayı1 = int(ilk_sayı)
    sayı2 = int(ikinci_sayı)
    print(sayı1, "/", sayı2, "=", sayı1 / sayı2)
except (ValueError, ZeroDivisionError) as hata:
    print("Bir hata oluştu!")
    print("original hata mesajı: ", hata)
```

~try, except, else

```
# -*- coding: utf-8 -*-  
  
try:  
    bolunen = int(input("bölünecek sayı: "))  
    bolen = int(input("bölen sayı: "))  
except ValueError:  
    print("Lütfen sadece sayı girin!")  
else:  
    try:  
        print(bolunen/bolen)  
    except ZeroDivisionError:  
        print("Bir sayıyı 0'a bölemezsiniz!")
```

~try, except, finally

```
try:
    dosya = open("dosyaadı", "r")
    ...burada dosyayla bazı işlemler yapıyoruz...
    ...ve ansızın bir hata oluşuyor...
except IOError:
    print("bir hata oluştu!")
finally:
    dosya.close()
```

~Dosya yazma

```
f=open("yapilacaklar.txt","w")  
f.write("aglamak istiyorum ")  
f.close()
```

~Dosya Okuma

```
f=open("/home/decoder/yapilacaklar.txt","r")  
print(f.readline())
```



Teşekkürler