Print KOMUTU

Ekrana yazdırılmak istenen ifade (" ") sembolleri içerisine yazılır.
\n kendinden sonraki ifadeyi alt satıra geçirir.
\t kendinden sonraki ifadeden önce tab kadar boşluk bıraktırır

In [ ]:
```
```

In [46]:
```
print("Merhaba")
print ("Merhaba Sevgili Öğrenciler")
print ("Merhaba \nOkulumuza Hoşgeldiniz")
print ("Program Adı: \tPython")
```

```
Merhaba
Merhaba Sevgili Öğrenciler
Merhaba
Okulumuza Hoşgeldiniz
Program Adı:    Python
```

Print - Format KOMUTU

.format ile dinamik yazdırma işlemi gerçekleştirilebilir.
Değişkenin belirtileceği dinamik kısım için {} sembolü kullanılır.

In [50]:
```
print("Benim Adım {}".format ('Erdem'))
print("Benim Adım {}, yaşım {}".format ('Erdem',32))
print("Benim Adım {0}, yaşım {1}".format ('Erdem',32))
print("Benim Adım {1}, yaşım {0}".format ('Erdem',32))
print("Benim Adım {ad}, yaşım {yas}".format (ad='Erdem', yas=32))
isim="Onur"
print("Benim Adım {}".format (isim))
```

```
Benim Adım Erdem
Benim Adım Erdem, yaşım 32
Benim Adım Erdem, yaşım 32
Benim Adım 32, yaşım Erdem
Benim Adım Erdem, yaşım 32
Benim Adım Onur
```

DEĞİŞKENLER:

Kabul Edilmeyen Değişken İsim Atama Senaryoları:
-Sayı ile başlamak
-Boşluk içermek
-"*/(gibi semboller içermek
-rezerve edilmiş isimleri kullanmak

In [83]:
```
sayi=10
```

```
print(sayi)
```

```
10
```

In [85]:
```
sayi=11
print(sayi)
```

```
11
```

In [87]:
```
sayi=sayi+1
print(sayi)
```

```
12
```

In [91]:
```
1sayi=10
print(1sayi)
```

```
  Cell In[91], line 1
    1sayi=10
    ^
SyntaxError: invalid decimal literal
```

In [93]:
```
yeni sayi=10
print(yeni sayi)
```

```
  Cell In[93], line 1
    yeni sayi=10
         ^
SyntaxError: invalid syntax
```

In [95]:
```
*sayi=10
print(*sayi)
```

```
  Cell In[95], line 1
    *sayi=10
    ^
SyntaxError: starred assignment target must be in a list or tuple
```

In [97]:
```
_sayi=10
print(_sayi)
```

```
10
```

VERİ TİPLERİ VE SAYI TİPLERİ

Tip ##### Kısaltma ##### Örnek

integer : ##### int ##### 5, 18
float : ##### float ##### 2.5 , 34.7
string : ##### str ##### 'abcd'
boolean : ##### bool ##### True, False
list : ##### list ##### [1, 2,True,'a', 1, True]
set : ##### set ##### {1,2,True, 'a'}
tuple: ##### tup ##### (1,2,True)
dictionary : ##### dict ##### {'isim': 'Mesut', 'yas' : 32}

int: tam sayılar
float: ondalık sayılar

In [111…
```
type(5)
```
Out[111… int

In [113…
```
type(12.4)
```
Out[113… float

In [115…
```
3+4
3*4
4-3
5%3
```
Out[115… 2

In [117…
```
3+4
```
Out[117… 7

In [119…
```
3*4
```
Out[119… 12

In [121…
```
4-3
```
Out[121… 1

In [123…
```
5%3
```
Out[123… 2

In [125…
```
2+6*3
```
Out[125… 20

In [127…
```
4.3*2
```
Out[127… 8.6

STRING VE CHAR

STRING: Metinleri ifade eder. ' ' ya da " " içerisinde gösterilir.
Metindeki ilk karakterin indeks numarası 0'dır.

In [9]:
```
Metin="Python"
print(Metin)
```

```
Python
```

In [16]:
```
Metin[0]
```

Out[16]:
```
'P'
```

In [18]:
```
Metin[2]
```

Out[18]:
```
't'
```

In [20]:
```
Metin[-1]
```

Out[20]:
```
'n'
```

In [22]:
```
Metin[-4]
```

Out[22]:
```
't'
```

[ ] Tek bir eleman alınır
[:] Başlangıç ve bitiş arasındaki eleman alınır

PYTHON

In [27]:
```
Metin[2:]
```

Out[27]:
```
'thon'
```

In [34]:
```
Metin[:3]
```

Out[34]:
```
'Pyt'
```

In [36]:
```
Metin[1:3]
```

Out[36]:
```
'yt'
```

-UZUNLUK: Len komutu String tipindeki değişkenin uzunluğunu verir.
-BAĞLAMA: + işareti ile iki stringi birleştirebiliriz.
-YENİ DEĞER ATAMA: = işareti ile değişkenin adına yeni bir değer atayabiliriz.
-TEKRARLAMA: * işareti işe değişkeni istediğimiz sayı kadar yazdırabiliriz.
-UPPER (harfleri büyütme), LOWER (harfleri küçültme), SPLIT (ayrıştırma) komutları kullanımı

In [129…
```
Metin="Python"
print(Metin)
```

```
Python
```

In [131…
```
len(Metin)
```

Out[131…
```
6
```

In [133…
```
Metin=Metin+ " ogren!"
print(Metin)
```
```
Python ogren!
```

In [135…
```
Metin*5
```
Out[135…
```
'Python ogren!Python ogren!Python ogren!Python ogren!Python ogre
n!'
```

In [137…
```
print(Metin.upper())
print(Metin.lower())
print(Metin.split())
print(Metin.split("o"))
```
```
PYTHON OGREN!
python ogren!
['Python', 'ogren!']
['Pyth', 'n ', 'gren!']
```

BOOLEAN

True ve False için " " kullanılmaz!

Boolean sorgulamalarda çalışır

Eşitlik sorgulamak için == ifadesi kullanılmalıdır.

In [5]:
```
deneme= True
sonuc= False
print(type(deneme))
deneme2= "True"
print(deneme2)
```
```
<class 'bool'>
True
```

In [13]:
```
yas1=18
yas2=20
print(yas2>18)
```
```
True
```

In [9]:
```
yas1==yas2
```
Out[9]:
```
False
```

In [7]:
```
yas1!=yas2
```
Out[7]:
```
True
```

In [15]:
```
not yas1==yas2
```
Out[15]:
```
True
```

LİSTE VE SET

Liste [ ] ile belirtilir.
Append >>> listeye ekleme yapar.
Pop >>> listeden çıkarma yapar (indeks değeri belirtilmezse son nesneyi çıkarır).
Sort >>> sıralama
Set >>> tekrar eden sayıları çıkarır

[a , b , c , d , e , a]
0 , 1 , 2 , 3 , 4 , 5

In [25]:
```
liste= ['a', 'b', 'c', 'd', 'e', 'a']
print(liste)
```

```
['a', 'b', 'c', 'd', 'e', 'a']
```

In [27]:
```
liste=liste+f
liste
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[27], line 1
----> 1 liste=liste+f
      2 liste

NameError: name 'f' is not defined
```

In [37]:
```
liste=liste+['f']
liste
```

Out[37]:
```
['a', 'b', 'c', 'd', 'e', 'a', 'f']
```

In [39]:
```
liste[3:5]
```

Out[39]:
```
['d', 'e']
```

In [45]:
```
liste.append('g')
liste
```

Out[45]:
```
['a', 'b', 'c', 'd', 'e', 'a', 'f', 'g']
```

In [47]:
```
liste.pop()
```

Out[47]:
```
'g'
```

In [49]:
```
liste
```

Out[49]:
```
['a', 'b', 'c', 'd', 'e', 'a', 'f']
```

In [55]:
```
liste.pop(5)
print(liste)
```

```
['a', 'b', 'c', 'd', 'e', 'f']
```

In [63]:
```
sayılar=[288, 35, 8791, 6, 17, 2, 2]
sayılar.sort()
```

In [65]:
```
sayılar
```

Out[65]:
```
[2, 2, 6, 17, 35, 288, 8791]
```

In [67]:
```
sayılar.reverse()
sayılar
```

Out[67]:
```
[8791, 288, 35, 17, 6, 2, 2]
```

In [69]:
```
set(sayılar)
```

Out[69]:
```
{2, 6, 17, 35, 288, 8791}
```

TUPLE

( ) ile tanımlanır
Veri setinin değişimine izin vermez
Count ( ) >>> kaç tane o nesneden olduğunu gösterir
index ( ) >>> kaçıncı indexte olduğunu gösterir
[a , b , c , d , e , a]
0 , 1 , 2 , 3 , 4 , 5

In [7]:
```
liste= ['a', 'b', 'c', 'd', 'e', 'a']
print(liste)
tup=('a', 'b', 'c', 'd', 'e', 'a')
print(tup)
```

```
['a', 'b', 'c', 'd', 'e', 'a']
('a', 'b', 'c', 'd', 'e', 'a')
```

In [11]:
```
liste[0]=123
liste
```

Out[11]:
```
[123, 'b', 'c', 'd', 'e', 'a']
```

In [21]:
```
tup[0]= 123
tup
```

```
---------------------------------------------------------------------
-------
TypeError                                 Traceback (most recent cal
l last)
Cell In[21], line 1
----> 1 tup[0]= 123
      2 tup

TypeError: 'tuple' object does not support item assignment
```

In [35]:
```
tup.count ('a')
```

Out[35]: 2

In [29]:
```
tup.count (True)
```

Out[29]: 0

In [33]:
```
tup.index('a')
```

Out[33]: 0

In [37]:
```
tup.index(True)
```

```
---------------------------------------------------------------------
-------
ValueError                                Traceback (most recent cal
l last)
Cell In[37], line 1
----> 1 tup.index(True)

ValueError: tuple.index(x): x not in tuple
```

DICTIONARY

{'saklanmak istenen bilgiler burada tutulur ve belirlenen anahtar kelimelerle daha sonra çağırılabilir' : 'bilgi' }

Keys, values, items komutları ile anahtarlar ve değişkenler çağırılabilir

In [9]:
```
dict1={'isim' : 'Erdem', 'yas': 32, 'lokasyon' : 'Manisa'}
dict1
```

Out[9]: {'isim': 'Erdem', 'yas': 32, 'lokasyon': 'Manisa'}

In [31]:
```
dict2= {
    'isim' : 'Erdem',
    'yas': 32,
    'lokasyon' :
    {
    'Doğdugu şehir' : 'İstanbul',
    'Yaşadığı şehir': 'Manisa'
    }
```

```
}
dict2
```

Out[31]:
```
{'isim': 'Erdem',
 'yas': 32,
 'lokasyon': {'Doğdugu şehir': 'İstanbul', 'Yaşadığı şehir': 'Mani
sa'}}
```

In [33]:
```
dict1['isim']
```

Out[33]:
```
'Erdem'
```

In [37]:
```
dict2['lokasyon']['Yaşadığı şehir']
```

Out[37]:
```
'Manisa'
```

In [43]:
```
dict1.keys()
```

Out[43]:
```
dict_keys(['isim', 'yas', 'lokasyon'])
```

In [45]:
```
dict1.values()
```

Out[45]:
```
dict_values(['Erdem', 32, 'Manisa'])
```

In [47]:
```
dict1.items()
```

Out[47]:
```
dict_items([('isim', 'Erdem'), ('yas', 32), ('lokasyon', 'Manis
a')])
```

In [ ]:
```

```