

**Python programlama 401 eğitim notları**  
**21.10.20**

**Eğitmen:M.Vahit Keskin**

**Anıl Nebi Şentürk**

**#turkcellgelecegiyazanlar**

Nesne yönelimli programlama('optimizasyon'):

1) sınıflara giriş ve sınıf tanımlamak

**Sınıf:** Benzer özellikler,ortak amaçlar taşıyan,içerisinde method ve değişkenler olan yapıdır.

```
class veribilimci():  
    print("Bu bir sınıftır")
```

**Sınıf Özellikleri(class attributes):**

```
1  #nesne yonemli programlama  
2  #sinif nedir?  
3  class veribilimci():  
4      bölüm=""  
5      sql="Evet"  
6      deneyim_yılı=0  
7      bildiği_diller=[]  
8  
9  #sınıfların özelliklerini erişme  
10 veribilimci.bölüm  
11 veribilimci.sql  
12  
13 #sınıf özelliklerini değiştirme  
14 veribilimci.sql="Hayır"  
15 veribilimci.sql
```

## Sınıf Örneklemesi: (instantiation)

```
#sınıf örneklendirmesi
ali=veribilimci
ali.sql
ali.bölüm
ali.deneyim_yılı
ali.bildiği_diller.append("python")
ali.bildiği_diller
veli=veribilimci
veli.sql
```

Console 1/A

```
In [19]: veli=veribilimci
In [20]: veli.sql
Out[20]: 'Hayır'
```

## Örnek Özellikleri:

```
class veribilimci():
    bölüm=""
    sql=""
    deneyim_yılı=0
    def __init__(self):
        self.bildiği_diller=[]
        self.bölüm=""
ali=veribilimci
ali.bildiği_diller
veli=veribilimci
veli.bildiği_diller
ali.bildiği_diller.append("python")
ali.bildiği_diller
veli.bildiği_diller.append("R")
```

## Örnek Methodları:

on 3.8)

```
Source Run Debug Consoles Projects Tools View Help
python 401.py
python_401.py* x python_401.2.part.py x

class veribilimci():
    def __init__(self):
        self.bildiği_diller=[]
        self.bölüm=""class veribilim-
        çalışanlar=[]
        bölüm=""
        bildiği_diller=""
        deneyim_y
        def dil_ekle(self,yeni_dil):
            self.bildiği_diller.appe
ali=veribilimci
ali.bildiği_diller
ali.bölüm
veli=veribilimci
veli.bildiği_diller
veli.bölüm
```

## MİRAS YAPILARI: (inheritance):

```
C:\Users\anils\python 401.py
temp.py x python_401.py* x python_401.2.part.py x

30         self.programming=""
31
32     class marketing():
33         def __init__(self):
34             self.storytelling=""
35         veribilimci1=datascience()
36         veribilimci1.
37     class employees_yeni():
38         def __init__(self,firstname,1
39             self.Firstname=Firstname
40             self.lastname=lastname
41             self.address=address
42     ali="employee_yeni"("a","b","c")
43     ali.firstname
44
```

***Fonksiyonel Programlamaya Giriş:*** - Birinci sınıf nesnelerdir.

-Yan etkisiz fonksiyonlardır. (pure function)/girdi ;çıkıtı

-Yüksek seviye fonksiyonlar

-Vektörel operasyonlar

### **YAN ETKİSİZ FONKSİYONLAR ÖRNEK 1 :**

```
46 a=90
47 def impure_sum(b):
48     return b+a
49 def pure_sum(a,b):
50     return a+b
51 impure_sum(6)
52 pure_sum(3,4)
```

### **YAN ETKİSİZ FONKSİYON ÖRNEK 2:**

```
7
8 def read(filename):
9     with open(filename,"r") as f:
10         return [line for line in f]
11     def count(lines):
12         return len(lines)
13     example_lines=read('deneme.txt')
14     lines_count= count(example_lines)
15     lines_count
```

## İSİMSİZ FONSKİYONLAR:

```
16
17 def old_sum(a,b):
18     return a+b
19 old_sum(4,5)
20 new_sum=lambda a,b:a+b
21 sirasiz_liste=[('b',3),('a',8),('d',12),('c',1)]
22 sirasiz_liste
23 sorted(sirasiz_liste,key=lambda x
24
25
```

```
Console I/A
('d',12),('c',1)]

In [97]: sorted(sirasiz_liste,key=lambda
x:x[1])
Out[97]: [('c', 1), ('b', 3), ('a', 8),
('d', 12)]

In [98]:
```

## VEKTÖREL OPERASYONLAR:

```
2 a=[1,2,3,4]
3 b=[2,3,4,5]
4 ab=[]
5 range(0,len(a))
6 for i in range(0,len(a)):
7     ab.append(a[i]*b[i])
8 ab
9 import numpy as np
10 a=np.array([1,2,3,4])
11 b=np.array([2,3,4,5])
12 a*b
13
```

```
Console I/A
In [109]: a=np.array([1,2,3,4])

In [110]: b=np.array([2,3,4,5])

In [111]: a*b
Out[111]: array([ 2,  6, 12, 20])
```

## MAP Filter ve Reduce Fonksiyonları:

```
1 #map,filter,reduce fonksiyonlar
2 liste=[1,2,3,4,5]
3 for i in liste:
4     print(i+10)
5 list(map(lambda x:x+10,liste))
6 liste=[1,2,3,4,5,6,7,8,9,10]
7 list(filter(lambda x:x%2==0,liste))
8
9 from functools import reduce
10 liste=[1,2,3,4]
11 reduce(lambda a,b:a+b,liste)
12
```

```
Console I/A
No documentation available
Click anywhere in this tooltip for additional help

TypeError: list indices must be integers
or slices, not tuple

In [120]: reduce(lambda a,b:a+b,liste)
Out[120]: 55
```

```
#hatalar
a=10
b=0
a/b
try:
    print(a/b)
except ZeroDevisionError:
    print("payda 0 olmaz")
a=8
b="8"
a/b
try:
    print(a/b)
except ZeroDevisionError
    print("string ve sayı")
```

```
Console I/A
No documentation available

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

In [135]: print("string ve sayı")
string ve sayı

In [136]:
```

## HATALAR:

## BÖLÜM SONU DEĞERLENDİRMESİ:

