

# Python

Gürer Özen

[gurer@pardus.org.tr](mailto:gurer@pardus.org.tr)

Yüksek düşünceler için yüksek bir dil gerekir.

-- Aristophanes

Herşeyi optimize edersiniz, daima mutsuz kalırsınız.

-- Donald Knuth

Bir kodun satır sayısını ölçeceksek, bunu üretilen satırlar olarak değil, harcanan satırlar olarak ölçmeliyiz.

-- Edsger Dijkstra

# Veri tipleri

None	Yok	...
string	karakter dizisi	"metin"
int	tamsayı	42
long	uzun tamsayı	287837483493843938283474L
complex	karmaşık sayı	(-3+5j)
float	kayar noktalı sayı	127.9
tuple	demet	( "ali", 42, "istanbul", 1.72 )
list	liste	[ "ali", "veli", "yavuz" ]
dict	sözlük	{ "ali": 42, "veli": 25, "yavuz": 54 }
set	küme	set([1, 2, 3, 4])

# Söz dizimi

*“Belki de en sonunda, yalnızca, daha büyüklerini inşa etmek için isimleriyle çağrılan ufak modüller yazmaya başlayacağız; böylece kaynak kodunda yerel yapıyı belirtmek için ayraçlar yerine girintileme kullanmak olanaklı hale gelebilecek.”*

-- Donald E. Knuth, Computing Surveys, Vol 6 No 4, Dec. 1974

```
def hesapla(x, y):
```

```
    a = x*y
```

```
    if a > 30:
```

```
        print “sonuç otuzdan büyük, ikiye bölüyorum”
```

```
        a /= 2
```

```
    return a
```

```
# Deneyelim fonksiyonumuzu
```

```
print 'Deniyoruz...'
```

```
hesapla(3, 5)
```

```
print “”
```

```
Bu basit bir örnekti, girintileme, açıklamalar, karakter dizileri için “ ve ’  
kullanımını gösteriyor
```

```
“”
```

# Fonksiyonlar

```
def selamla():
```

```
    print "Merhaba Ali"
```

```
def selamla2(kisi, selam):
```

```
    print kisi, selam
```

```
def selamla3(kisiler, selam='Merhaba'):
```

```
    for kisi in kisiler:
```

```
        print selam, kisi
```

```
def selamla4(selam, *kisiler):
```

```
    for kisi in kisiler:
```

```
        print selam, kisi
```

```
def selamla5(*kisiler, **kwargs):
```

```
    selam = "Merhaba"
```

```
    if kwargs.has_key("selam"):
```

```
        selam = kwargs["selam"]
```

```
    selamla3(kisiler, selam)
```

# Döngüler

```
for eleman in [ 1, 2, 3, 4, 5 ]:  
    print eleman
```

```
for numara in range(5, 50):  
    print numara
```

```
for satir in file("/var/log/messages"):  
    print satir
```

```
for yol, dizinler, dosyalar in os.walk("/home/gurer"):  
    print yol, "dizinde şu dosyalar var:"  
    for dosya in dosyalar:  
        print dosya
```

```
def sayidizisi(bitis):  
    for i in range(1, bitis):  
        yield i * i
```

```
for sayi in sayidizisi(10):  
    print sayi
```

# Fonksiyonel Programlama Araçları

```
def fonk(x, y):  
    return x, y
```

```
lambda x, y: return x, y
```

```
dizi = ( 1, 2, 3, 4, 5, 6 )
```

```
map(lambda x: x*x, dizi)  
[1, 4, 9, 16, 25, 36]
```

```
filter(lambda x: x & 1, dizi)  
[1, 3, 5]
```

```
reduce(lambda x, y: x + y, dizi)  
21
```

```
[ 3 + x for x in dizi if x & 1 ]  
[4, 6, 8]
```

# Nesneler

Tüm veriler birer nesne.

```
a = "python"
```

```
dir(a)
```

```
['__class__', '__contains__', '__doc__', '__eq__', '__ge__', '__hash__', '__init__',  
'__len__', '__str__', 'capitalize', 'endswith', 'find', 'index', 'join', 'lower', 'lstrip',  
'replace', 'rfind', 'rindex', 'rsplit', 'rstrip', 'split', 'startswith', 'strip', 'upper' ...
```

```
a.__class__
```

```
<type 'str'>
```

```
a.upper
```

```
<built-in method upper of str object at 0xa7badc00>
```

```
help(a.upper)
```

```
Return a copy of the string S converted to uppercase.
```

```
a.upper()
```

```
"PYTHON"
```

# Nesneler

```
class Oyuncu:
    def __init__(self, isim):
        self.isim = isim
        self.puan = 0

    def puan_ekle(self, alinan_tas_sayisi):
        self.puan += alinan_tas_sayisi * 10

    def __str__(self):
        return "Oyuncu %s, %d puana sahip" % (self.isim, self.puan)

oyuncu = Oyuncu("Ali")
oyuncu.puan_ekle(3)
oyuncu.puan_ekle(5)
print oyuncu
Oyuncu Ali, 80 puana sahip
```

# Nesneler

```
class Pencere(QMainWindow):  
    def __init__(self):  
        QMainWindow.__init__(self)  
        ...  
        self.show()  
  
    @staticmethod  
    def pencere_goster():  
        ...
```

# Modüller

```
import socket  
s = socket.socket(socket.AF_UNIX, socket.SOCK_STREAM)
```

```
import os, time, sys
```

```
from os import environ
```

```
from os import *
```

```
yardimci.py:  
def kare(x):  
    return x*x
```

```
ana.py:  
import yardımci
```

```
print kare(5)
```

# Hatalar

```
try:  
    data = file("/home/gurer/lala").read()  
except IOError:  
    print "dosya yok"
```

```
try:  
    ...  
except Exception, e:  
    ...  
except:  
    ...
```

```
try:  
    ...  
finally:  
    ...
```

```
class MyException(Exception):  
    ...
```