listeler

- liste: sıralı değer kümesi
- öğe: liste elemanları
- liste X dizgi: her bir elemanın karakter olmak zorunda değil
- liste ve dizgi: dizi

9.1 Liste değerleri

PB:5643 ve 5644

```
>>> kelime = ["spam", "bungee", "swallow"]
>>> sayi = [10, 20, 30, 40]
>>> karisik = ["hello", 2.0, 5, True, [10, 20]]
>>> print sayi
[10, 20, 30, 40]
>>> print kelime
['spam', 'bungee', 'swallow']
>>> print karisik
['hello', 2.0, 5, True, [10, 20]]
```

öğelere erişme

- → öğelere erişim köşeli parantez- []
- → indis: tamsayı, pozitif/negatif, sıfır ilk elemandır
- → PB:5645

```
>>> sayi = [10, 20, 30, 40]
>>> sayi[0]
10
>>> sayi[2]
30
>>> sayi[5]
Traceback (most recent call last):
  File "<input>", line 1, in <module>
IndexError: list index out of range
>>> sayi[1.0]
Traceback (most recent call last):
  File "<input>", line 1, in <module>
TypeError: list indices must be integers, not float
>>> sayi[-1]
40
>>> sayi[-2]
30
```

döngü

→ liste dolaşımı (PB:5646)

```
>>> kelime = ["spam", "bungee", "swallow"]
>>> i = 0
>>> while i < 3:
...     print kelime[i]
...     i += 1
...
...
spam
bungee
swallow
```

liste boyutu

→ len: listedeki eleman sayısı (PB:5647)

```
>>> kelime = ["spam", "bungee", "swallow"]
>>> i = 0
>>> while i < len(kelime):
...     print kelime[i]
...     i += 1
...
...
spam
bungee
swallow
```

→ aşağıdaki liste kaç elemanlıdır?

```
['spam!', 1, ['Brie', 'Roquefort', 'Pol le Veq'], [1, 2, 3]]
```

üyelik

→ in: öğenin listede olup-olmadığını sınamamızı sağlar (PB:5648)

```
>>> kelime = ["spam", "bungee", "swallow"]
>>> "hum" in kelime
False
>>> "swallow" in kelime
True
>>> ne = "spam"
>>> ne in kelime
True
>>> "foo" not in kelime
True
```

liste işlemleri

→ +: birleştirme, *: tekrarlama (PB:5649)

```
>>> a = [1, 2, 3]
>>> b = [4, 5, 6]
>>> c = a + b
>>> print c
[1, 2, 3, 4, 5, 6]
>>>
>>> [0]*4
[0, 0, 0, 0]
>>> a * 4
[1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3]
```

dilimleme

→ dizgideki gibi (PB:5650)

```
>>> kelime
['spam', 'bungee', 'swallow']
>>> kelime[1:2]
['bungee']
>>> kelime[1:3]
['bungee', 'swallow']
>>> kelime[1:]
['bungee', 'swallow']
>>> kelime[:1]
['spam']
>>> kelime[:2]
['spam', 'bungee']
>>> kelime[:]
['spam', 'bungee', 'swallow']
```

range işlevi

→ range(start, stop, step) (PB:5651)

```
>>> range(10)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> range(5,10)
[5, 6, 7, 8, 9]
>>> range(5,15,3)
[5, 8, 11, 14]
>>> range(20,4,-5)
[20, 15, 10, 5]
>>> range(10,20,-2)
[]
```

değiştirilebilirlik

→ dizgi: değiştirilemez

→ liste: değiştirilebilir (PB:5653)

```
>>> kelime
['hum', 'bungee', 'bar']
>>> kelime[0] = "hum"
>>> kelime[-1] = "bar"
>>> kelime
['hum', 'bungee', 'bar']
>>>
>>> str = "TEST"
>>> str[2] = 'X'
Traceback (most recent call last):
  File "<input>", line 1, in <module>
TypeError: 'str' object does not support item assignment
>>>
>>> strlst = ['T', 'E', 'S', 'T']
>>> strlst[2] = 'X'
>>> strlst
['T', 'E', 'X', 'T']
>>>
>>> strlst[1:3] = ['.', ':']
>>> strlst
['T', '.', ':', 'T']
```

devam

devam

```
>>> strlst[1:1] = ['#', '*', '+']
>>> strlst
['T', '#', '*', '+', '.', ':', 'T']
>>>
>>> strlst[4:6] = []
>>> strlst
['T', '#', '*', '+', 'T']
```

öğe silme

→ del: öğeleri siler (PB:5654)

```
>>> kelime
['hum', 'bar']
>>> del kelime[1]
>>> kelime
['hum']
>>>
>>> num = range(10)
>>> del num[4:7]
>>> num
[0, 1, 2, 3, 7, 8, 9]
```

nesneler ve değerler

→ her nesne tekil tanımlayıcıya (identifier) sahiptir

→ id: nesnenin tanımlayıcısı

→ dizgi x liste: değiştirilebilme (PB:5655)

```
>>> a = "banana"
>>> b = "banana"
>>> id(a), id(b)
(155905856, 155905856)
>>>
>>> a = [1, 2, 3]
>>> b = [1, 2, 3]
>>> id(a), id(b)
(155899692, 155898220)
>>>
```

→ dizgi: aynı değer, aynı tanımlayıcı (unmutable)

→ liste: aynı değer, farklı tanımlayıcı (mutable)

takma isimler

- → değişkenler nesneleri gösterir
- → bir değişkeni bir başkasına atarsak, her iki değişken aynı nesneyi gösterir
- → burada b, a'nın takma ismidir (aliased)
- → takma isimlideki değişiklik diğerini etkiler (PB:5656)

```
>>> a = [1, 2, 3]
>>> b = a
>>> id(a) == id(b)
True
>>> b[0] = 99
>>> a
[99, 2, 3]
```

shallow X deep copy

→ shallow X deep copy (PB:5657)

```
>>> a = [1, 2, 3]
>>> b = a
>>> id(a) == id(b)
True
>>> b[0] = 99
>>>
>>> a = [1, 2, 3]
>>> b = a[:]
>>> id(a) == id(b)
False
>>> b[0] = 99
>>> b
[99, 2, 3]
>>> a
[1, 2, 3]
```

döngüler

→ liste elemanlarını while ile yazdırmak mümkün (PB:5660)

→ for ile daha kısaltmak

```
>>> kelimeler = ["spam", "foo", "bar", "yum"]
>>> i = 0
>>> while i < len(kelimeler):
...     print kelimeler[i]
...     i += 1
...
...
spam
foo
bar
yum
>>>
>>> for i in range(len(kelimeler)):
...     print kelimeler[i]
...
...
spam
foo
bar
yum
```

devam

```
>>> for kelime in kelimeler:
...     print kelime
...
...
spam
foo
bar
yum
```

döngüler

→ for + range

```
>>> for i in range(10):
...     if i%3 == 0:
...         print i, "uce tam bolunur"
...
...
...
0 uce tam bolunur
3 uce tam bolunur
6 uce tam bolunur
9 uce tam bolunur
```

→ enumarate işlevi

```
>>> for ind, kelime in enumerate(kelimeler):
...     print ind, ":", kelime
...
...
0 : spam
1 : foo
2 : bar
3 : yum
```

işleve parametre

→ işleve geçirilen listenin **referansıdır** (shallow, etiket, takma ismi)

→ işlev içerisindeki listede yapılan değişiklik, asıl listeyi etkiler

→ deep copy (PB:5661)

```
>>> from d09 import double_stuff
>>> things = [2, 5, 'Spam', 9.5]
>>> double_stuff(things)
>>> things
[4, 10, 'SpamSpam', 19.0]
>>>
>>> from d09 import double_stuff_v2
>>> things = [2, 5, 'Spam', 9.5]
>>> double_stuff_v2(things)
[4, 10, 'SpamSpam', 19.0]
>>> things
[2, 5, 'Spam', 9.5]
```

→ things'i illa da değiştirmek isterseniz işlevin dönüş değerini karşılat

içiçe listeler

→ PB:5662)

```
>>> nested = ["hello", 2.0, 5, [10, 20]]
>>> elem = nested[0]
>>> elem[0:3]
'hel'
>>> nested[3]
[10, 20]
>>> nested[3][1]
20
```

matrisler

→ PB:5663

```
>>> matrix = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
>>> matrix[1]
[4, 5, 6]
>>> matrix[1][2]
6
```

test güdümlü çalışma

→ otomatikleştirilmiş testlerle güdüle (d09.py)

```
1  def make_matrix(rows, columns):
2      """
3        >>> make_matrix(3, 5)
4        [[0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0]]
5      """
6
7  if __name__ == '__main__':
8      import doctest
9      doctest.testmod()
```

test

→ d09.py

```
1  def make_matrix_v2(rows, columns):
2      """
3          >>> make_matrix(3, 5)
4          [[0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0]]
5          >>> make_matrix(4, 2)
6          [[0, 0], [0, 0], [0, 0], [0, 0]]
7          >>> m = make_matrix(4, 2)
8        >>> m[1][1] = 7
9        >>> m
10       [[0, 0], [0, 7], [0, 0], [0, 0]]
11     """
12     matrix = []
13     for row in range(rows):
14         matrix += [[0] * columns]
15     return matrix
```

devam

→ test, PB:5664

```
>>> from d09 import make_matrix_v1
>>> m = make_matrix_v1(4, 3)
>>> m[1][2] = 99
>>> m
[[0, 0, 99], [0, 0, 99], [0, 0, 99], [0, 0, 99]]
>>>
>>> from d09 import make_matrix_v2
>>> m = make_matrix_v2(4, 3)
>>> m[1][2] = 99
>>> m
[[0, 0, 0], [0, 0, 99], [0, 0, 0], [0, 0, 0]]
```

dizgi <--> liste

→ PB:5665

→ dizgi ⟹ *liste*dönüşü*m*ü

→ str işlevi, tersinirleştirmez!

```
>>> cumle = "Bugun hava guzel"
>>> lst = list(cumle)
>>> lst
['B', 'u', 'g', 'u', 'n', ' ', \
'h', 'a', 'v', 'a', ' ', \
'g', 'u', 'z', 'e', 'l']
>>> str(lst)
"['B', 'u', 'g', 'u', 'n', ' ', \
'h', 'a', 'v', 'a', ' ', \
'g', 'u', 'z', 'e', 'l']"
>>> str(lst)[0]
'['
```

dizgi <--> liste

→ string modülü: split ve join

```
>>> import string
>>> kelimeler = string.split(cumle)
>>> kelimeler
['Bugun', 'hava', 'guzel']
>>>
>>> ycumle = string.join(kelimeler, ";")
>>> ycumle
'Bugun;hava;guzel'
```

dizgi <--> liste

→ dizgi: değiştirilemez, liste:değiştirilebilir

```
>>> dizgi = "TEST"
>>> dizgi[2] = 'X'
Traceback (most recent call last):
  File "<input>", line 1, in <module>
TypeError: 'str' object does not support item assignment
>>> lst = list(dizgi)
>>> lst[2] = 'X'
>>> lst
['T', 'E', 'X', 'T']
>>> ydizgi = string.join(lst, '')
>>> ydizgi
'TEXT'
```

sıra sizde

1. aşağıdaki listede dolaşan ve her öğenin boyutunu yazan döndü?

```
['spam!', 1, ['Brie', 'Roquefort', 'Pol le Veq'], [1, 2, 3]]
```

2. liste elemanı da listeyse (içiçe liste) onların her birini de yazdıran döngü

```
0: 'spam'
1: 1
2:0: 'Brie'
2:1: 'Roquefort'
2:2: 'Pol le Veq'
3:0: 1
3:1: 2
3:2: 3
```

sıra sizde

d09ss.py dosyası içeriği

sıra sizde

→ shallow X deep

```
1  this = ['I', 'am', 'not', 'a', 'crook']
2  that = ['I', 'am', 'not', 'a', 'crook']
3  print "Test 1: %s" % (id(this) == id(that))
4  that = this
5  print "Test 2: %s" % (id(this) == id(that))
```

sıra sizde

→ listeler üzerinde dört işlem: d09list.py

→ stringler üzerinde işlem: d09str.py