Python ile Programlamaya Giriş

DERS 7: LİSTELER

DR. HÜSEYİN BAHTİYAR

CC BY

Programlama

- Algoritma
  - Bir problemi çözmek için kullanılan kurallar.
- Veri Yapısı
  - Bilgisayardaki verileri organize etmek için kullanılan belirli yöntemler.

https://en.wikipedia.org/wiki/Algorithm
https://en.wikipedia.org/wiki/Data_structure

Toplu veri "collection" ne değildir

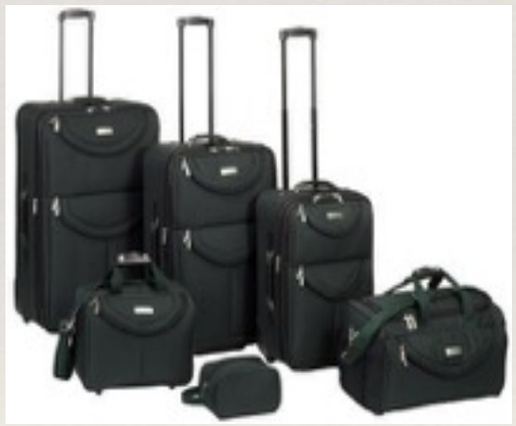
- Bir çok değişkende değişken bir değerlidir. Değişkene yeni atandığında eski değerin üzerine yazılır.

```
$ python
>>> x = 2
>>> x = 4
>>> print(x)
4
```

Bir List Collection gibidir

- collection tek bir değişkene birden fazla değer atamaya izin verir.

- collection kullanışlıdır çünkü bir çok değeri tek bir paket ile istediğimiz yere taşıyabiliriz.



```
friends = [ 'Joseph', 'Glenn', 'Sally' ]

tasinacak = [ 'corap', 'tshirt', 'parfum' ]
```

Liste Sabitleri

- List sabitleri [ ] işaretleri arasında yazılır ve birbirlerinden virgül ile ayrılırlar.

- List elemanı herhangi bir python objesi olabilir, farklı liste bile olabilir.

- List içi boş olabilir.

```
>>> print([1, 24, 76])
[1, 24, 76]
>>> print(['red', 'yellow', 'blue'])
['red', 'yellow', 'blue']
>>> print(['red', 24, 98.6])
['red', 24, 98.6]
>>> print([ 1, [5, 6], 7])
[1, [5, 6], 7]
>>> print([])
[]
```

Döngülerde Liste Kullandık mı?

```
for i in [5, 4, 3, 2, 1] :
    print(i)
print('Ateş!')
```

```
5
4
3
2
1
Ateş!
```

Liste ve Belirli Döngüler
Çok iyi arkadaşlar

```
friends = ['Joseph', 'Glenn', 'Sally']
for friend in friends :
    print('Happy New Year:', friend)
print('Done!')
```

```
Happy New Year: Joseph
Happy New Year: Glenn
Happy New Year: Sally
Done!
```

```
z = ['Joseph', 'Glenn', 'Sally']
for x in z:
    print('Happy New Year:', x)
print('Done!')
```

Listenin içine bakmak

Dikdörtgen parantezleri [ ] kullanarak listenin istediğimiz elemanına erişebiliriz.

Joseph	Glenn	Sally
0	1	2

```
>>> friends = [ 'Joseph', 'Glenn', 'Sally' ]
>>> print(friends[1])
Glenn
>>>
```

Listeler Değiştirilebilir

Listenin herhangi bir elemanını listenin indis elemanını kullanarak değiştirebiliriz

```
>>> fruit = 'Banana'
>>> fruit[0] = 'b'
Traceback
TypeError: 'str' object does not
support item assignment
>>> x = fruit.lower()
>>> print(x)
banana
>>> lotto = [2, 14, 26, 41, 63]
>>> print(lotto)
[2, 14, 26, 41, 63]
>>> lotto[2] = 28
>>> print(lotto)
[2, 14, 28, 41, 63]
```

Listenin uzunluğu?

- Listeler de dahil olmak üzere herhangi bir sıralı verinin uzunluğunu len() komutunu kullanarak bulabiliriz.

- Kullanımı len(degiskenadi)

```
>>> greet = 'Hello Bob'
>>> print(len(greet))
9
>>> x = [ 1, 2, 'joe', 99]
>>> print(len(x))
4
>>>
```

range fonksiyonu

- range fonksiyonu 0 ile list numarası arasında numara döndürür.
- Böylece bizler de döngüler için indis oluşturabiliriz (ve önceki örneklerde oluşturduk.)

```
>>> print(range(4))
[0, 1, 2, 3]
>>> friends = ['Joseph', 'Glenn', 'Sally']
>>> print(len(friends))
3
>>> print(range(len(friends)))
[0, 1, 2]
>>>
```

Döngü örnekleri

```
friends = ['Joseph', 'Glenn', 'Sally']

for friend in friends :
    print('Happy New Year:', friend)

for i in range(len(friends)) :
    friend = friends[i]
    print('Happy New Year:', friend)
```

```
Happy New Year: Joseph
Happy New Year: Glenn
Happy New Year: Sally
```

```
>>> friends = ['Joseph', 'Glenn', 'Sally']
>>> print(len(friends))
3
>>> print(range(len(friends)))
[0, 1, 2]
>>>
```

+ operatörü ile Listeleri bağlamak

İki listeyi birbirine ekleyerek yeni bir liste oluşturabiliriz.

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

Listeleri : ile parçalamak

```
>>> t = [9, 41, 12, 3, 74, 15]
>>> t[1:3]
[41,12]
>>> t[:4]
[9, 41, 12, 3]
>>> t[3:]
[3, 74, 15]
>>> t[:]
[9, 41, 12, 3, 74, 15]
```

Uyarı! Listeleri parçalayarak küçük listeler oluştururken : sağında kalan indise kadar böler (son indis dahil olmaz)

List komutları

- dir komutu ile herhangi bir method için kullanılabilecek komutları görebiliriz.

```
>>> x = list()
>>> type(x)
<type 'list'>
>>> dir(x)
['append', 'count', 'extend', 'index', 'insert', 'pop', 'remove', 'reverse',
'sort']
>>>
```

http://docs.python.org/tutorial/datastructures.html

Liste yaratmak

- Boş bir liste oluşturup listenin içini append komutu ile istediğimiz elemanlar ile doldurabiliriz.

- append komutu ile eklenen değişken listenin sonuna eklenir böylece sıra bozulmaz.

```
>>> stuff = list()
>>> stuff.append('book')
>>> stuff.append(99)
>>> print(stuff)
['book', 99]
>>> stuff.append('cookie')
>>> print(stuff)
['book', 99, 'cookie']
```

Listede var mı?

- Python'da aradığımız değişkenin listede olup olmadığının bilgisinin veren iki operatör bulunmaktadır
- Bunlar maktıksal operatörlerdir ve Doğru veya Yanlış sonucu döndürürler (True veya False)
- Listenin içeriğini değiştirmezler

```
>>> some = [1, 9, 21, 10, 16]
>>> 9 in some
True
>>> 15 in some
False
>>> 20 not in some
True
>>>
```

Listeyi Sıralamak

- Liste bir çok değişkeni tutabilir ve biz değiştirene kadar aynı şekilde tutar
- Listeler sıralanabilir.
- Listeleri sıralamak için sort() fonksiyonu kullanılır.

```
>>> friends = [ 'Joseph', 'Glenn', 'Sally' ]
>>> friends.sort()
>>> print(friends)
['Glenn', 'Joseph', 'Sally']
>>> print(friends[1])
Joseph
>>>
```

Hazır Fonksiyonlar

- Python içerisinde listeyi parametre olarak kullanan bir çok hazır fonksiyon bulunmaktadır.
- Döngülerde yaptığımız işleri çok daha basit bir şekilde yapabiliriz :)

```
>>> nums = [3, 41, 12, 9, 74, 15]
>>> print(len(nums))
6
>>> print(max(nums))
74
>>> print(min(nums))
3
>>> print(sum(nums))
154
>>> print(sum(nums)/len(nums))
25.6
```

Ortalama

```
total = 0
count = 0
while True :
    inp = input('Sayı gir: ')
    if inp == 'done' : break
    value = float(inp)
    total = total + value
    count = count + 1

average = total / count
print('Ort:', average)
```

```
Sayı gir: 3
Sayı gir: 9
Sayı gir: 5
Sayı gir: done
Average: 5.66666666667
```

```
numlist = list()
while True :
    inp = input('Sayı gir: ')
    if inp == 'done' : break
    value = float(inp)
    numlist.append(value)

average = sum(numlist) / len(numlist)
print('Ort:', average)
```

String ve Listenin arkadaşlığı

```
>>> abc = 'With three words'
>>> stuff = abc.split()
>>> print(stuff)
['With', 'three', 'words']
>>> print(len(stuff))
3
>>> print(stuff[0])
With
```

```
>>> print(stuff)
['With', 'three', 'words']
>>> for w in stuff :
...     print(w)
...
With
Three
Words
>>>
```

Split fonksiyonu stringi böler ve daha küçük string parçaları oluşturur böylece erişmek istediğimiz özel bir kelime veya döngüye erişebiliriz.

```
>>> line = 'A lot          of spaces'
>>> etc = line.split()
>>> print(etc)
['A', 'lot', 'of', 'spaces']
>>>
>>> line = 'first;second;third'
>>> thing = line.split()
>>> print(thing)
['first;second;third']
>>> print(len(thing))
1
>>> thing = line.split(';')
>>> print(thing)
['first', 'second', 'third']
>>> print(len(thing))
3
>>>
```

Acknowledgements / Contributions

These slides are Copyright 2010- Charles R. Severance (www.dr-chuck.com) of the University of Michigan School of Information and open.umich.edu and made available under a Creative Commons Attribution 4.0 License. Please maintain this last slide in all copies of the document to comply with the attribution requirements of the license. If you make a change, feel free to add your name and organization to the list of contributors on this page as you republish the materials.

Initial Development: Charles Severance, University of Michigan School of Information

… Insert new Contributors and Translators here