

Programmeerimine

9. loeng

Täna loengus

- Sõnastikud
- Eeldefineeritud meetodeid sõnastikel
- Hulgad
- Eeldefineeritud meetodeid hulkadel
- Kahekordsed tsüklid ühemõõtmelisel järjendil

Sõnastikud

Sõnastikud - motivatsioon

- Telefoniraamat

Näide

```
names = ['Alice', 'Betti', 'Anu', 'Eliise', 'Annely']
```

Sõnastikud - motivatsioon

- Telefoniraamat

Näide

```
names = ['Alice', 'Betti', 'Anu', 'Eliise', 'Annely']
```

```
numbers = ['2341', '9102', '3158', '0142', '5551']
```

Sõnastikud - motivatsioon

- Telefoniraamat

Näide

```
names = ['Alice', 'Betti', 'Anu', 'Eliise', 'Annely']
```

```
numbers = ['2341', '9102', '3158', '0142', '5551']
```

```
nr = numbers[names.index('Eliise')]
```

Sõnastikud - motivatsioon

- Telefoniraamat

Näide

```
names = ['Alice', 'Betti', 'Anu', 'Eliise', 'Annely']
```

```
numbers = ['2341', '9102', '3158', '0142', '5551']
```

```
nr = numbers[names.index('Eliise')]
```

- Lihtsam oleks:

```
nr = phonebook['Eliise']
```

Sõnastikud

- **Sõnastik** (ingl. *dictionary*) on dünaamilise suurusega mu-teeritav kogumandmestruktuur, kus elemente identifitseeri-takse **võtmetega**.
- Esitatakse (võti:väärtus)-paaride loendina loogeliste sulgude vahel.

$\{ key_1 : expr_1, key_2 : expr_2, \dots, key_n : expr_n \}$

- Võtmed key_i võivad olla suvalist *mittemuteeritavat* tüüpi väärtused.
- Kõik võtmed key_i on reeglina sama tüüpi ja ka väärtused $expr_i$ on reeglina ühte tüüpi, kuid Python lubab ka erinevat tüüpi võtmeid ja/või väärtusi.

Sõnastikud

- Sarnaselt järjenditega, toimub sõnastiku elementidele juurdepääs indekseerimisega, kuid täisarvuliste indeksite asemel kasutatakse võtmeid.
- Samuti saab üle sõnastiku itereerimiseks kasutada for-tsükli.

Näide

```
phonebook = {'Alice': '2341',  
             'Betti': '9102',  
             'Anu': '3158',  
             'Eliise': '0142',  
             'Annely': '5551'}  
  
for nimi in phonebook:  
    print(nimi + '\t' + phonebook[nimi])
```

Sõnastikud

Näide

```
table = {'Python': 'Guido van Rossum',  
         'Perl':   'Larry Wall',  
         'Tcl':    'John Ousterhout' }
```

```
for lang in table:  
    print(lang, '\t', table[lang])
```

Sõnastikud

Eeldefineeritud funktsioone ja meetodeid

Meetod

`k in D`

`D.keys()`

`D.values()`

`D.items()`

`D.copy()`

`D.pop(k)`

`del D[k]`

`len(D)`

Kirjeldus

kas võti `k` on sõnastikus?

tagastab võtmete dünaamilise vaate

tagastab väärtuste dünaamilise vaate

tagastab (võti,väärtus)-paaride dünaamilise vaate

tagastab sõnastiku koopia

tagastab indeksiga `k` väärtuse

ja eemaldab selle sõnastikust

eemaldab indeksiga `k` väärtuse sõnastikust

tagastab sõnastiku suuruse

Sõnastikud

Näide

```
D1 = {'John':23, 'Anne':18}
a = list(D1.keys())      # a = ['Anne', 'John']
b = list(D1.values())    # b = [18, 23]
c = list(D1.items())     # c = [('Anne', 18), ('John', 23)]
x = D1.pop('John')       # x = 23; D1 = {'Anne':18}
D1['James'] = 20          # D1 = {'James': 20, 'Anne': 18}
D1['Anne'] = 19           # D1 = {'James': 20, 'Anne': 19}
D2 = D1.copy()           # D2 = {'James': 20, 'Anne': 19}
D2['Anne'] = 20           # D2 = {'James': 20, 'Anne': 20}
del D2['James']           # D2 = {'Anne': 20}
                          # D1 = {'James': 20, 'Anne': 19}
```

Näide: sõnade esinemissagedused

Näide

```
def sagedused(fail):  
    f = open(fail)  
    d = {}  
    for line in f:  
        for w in line.split():  
            w = w.strip(' !?.,;: ')  
            w = w.lower()  
            if w != '':  
                if w in d:  
                    d[w] += 1  
                else:  
                    d[w] = 1  
    f.close()  
    return d
```

Näide - Väike andmebaas

```
people = {'Alice': {'phone': '2341',  
                    'addr': 'Foo drive 23'},  
          'Betty': {'phone': '9102',  
                    'addr': 'Bar street 42'},  
          'Cecil': {'phone': '3158',  
                    'addr': 'Baz avenue 90'}}  
  
labels = {  
    'phone': 'phone number',  
    'addr': 'address'}  
  
name = input('Name: ')  
request = input('Phone number (p) or address (a)? ')  
if request == 'p': key = 'phone'  
if request == 'a': key = 'addr'  
if name in people:  
    print(name, labels[key], ': ', people[name][key])
```

Hulgad

Hulgad

- **Hulk** (ingl. *set*) on mittekorduvate järjestamata elementidega kogumandmestruktuur.
- Hulki on kahte tüüpi: **set** - muteeritav ja **frozenset** - mittemuteeritav.

```
s1 = {1, 2, 3, 4, 5}
{1, 2, 3, 4, 5}
s2 = set('Hello, World!')
{'H', 'l', 'o', '!', ' ', 'e', 'd', 'r', 'W', ',', ' '}
```

Hulgad

Hulga läbivaatamine

```
s1 = set('abracadabra')  
for el in s1:  
    print(el, ' ', end = '')
```

```
# Trükiib:  
r d b c a
```

Tehted hulkadega

<i>Matemaatiline märk</i>	<i>Pythoni märk</i>	<i>Kirjeldus</i>
$\in$	<code>in</code>	on hulga element
$\notin$	<code>not in</code>	ei ole hulga element
$=$	<code>==</code>	on võrdne
$\neq$	<code>!=</code>	mittevõrdne
$\subset$	<code><</code>	on range alamhulk
$\subseteq$	<code><=</code>	on alamhulk
$\supset$	<code>></code>	on range ülemhulk
$\cap$	<code>&</code>	ühisosa
$\cup$	<code> </code>	ühend
$-$	<code>-</code>	vahe
Δ	<code>^</code>	sümmeetiline vahe

Tehted hulkadega

Näiteid

```
s1 = set(range(5))           # {0, 1, 2, 3, 4}
s2 = set(range(2, 4))        # {2, 3}
s3 = set(range(6, 8))        # {6, 7}
s4 = {2, 3, -1, 5}

print(3 in s1)                # Trükkib: True
print(2 not in s1)            # Trükkib: False
print(s1 == s2)               # Trükkib: False
print(s2 < s1)                # Trükkib: True
print(s1 & s2)                 # Trükkib: {2, 3}
print(s1 | s3)                # Trükkib: {0, 1, 2, 3, 4, 6, 7}
print(s1 ^ s4)                # Trükkib: {0, 1, 4, 5, -1}
```

Hulgad

Eeldefineeritud funktsioone ja meetodeid

Meetod

S.add(el)

S.update(S1)

S.remove(el)

S.pop()

S.clear()

S.copy()

Kirjeldus

lisab elemendi *el* hulka *S* (juhul kui seda varem polnud)

täiendab hulka *S* teise hulga *S1* elementidega

eemaldab elemendi *el* hulgast *S*

eemaldab ja tagastab hulgast *S* suvalise elemendi

tühjendab hulga *S*

tagastab hulga *S* koopia (shallow)

Tsüklid üksteise sees

Tsüklid üksteise sees

- Kui üks tsükkel on teise sees, siis sisemine tsükkel täidetakse täielikult välimise tsükli iga tsükliammu korral
- Kõikide iteratsioonide arvu saamiseks tuleb korrutada kõikide tsüklite korduste arvud omavahel.

Tsüklid üksteise sees

Näide - Tunnid, minutid, sekundid

```
for tunnid in range(0,24):  
    for minutid in range(0,60):  
        for sekundid in range(0,60):  
            print (tunnid, ":", minutid, ":", sekundid)
```

Suur tänu osalemast

ja

kohtumiseni!