Programmeerimine

11. loeng

Täna loengus

- Sõnastikud
- Eeldefineeritud meetodeid sõnastikel
- Hulgad
- Eeldefineeritud meetodeid hulkadel

Sõnastikud

Sõnastikud - motivatsioon

- Ülesanne
  On antud tekst. Lugeda kokku, mitu korda iga täht selles esineb.

Sõnastikud - motivatsioon

- Ülesanne
  On antud tekst. Lugeda kokku, mitu korda iga täht selles esineb.

Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twice she had peeped into the book her sister was reading, but it had no pictures or conversations in it, 'and what is the use of a book,' thought Alice 'without pictures or conversations?'

So she was considering in her own mind (as well as she could, for the hot day made her feel very sleepy and stupid), whether the pleasure of making a daisy-chain would be worth the trouble of getting up and picking the daisies, when suddenly a White Rabbit with pink eyes ran close by her.

There was nothing so *very* remarkable in that; nor did Alice think it so *very* much out of the way to hear the Rabbit say to itself, 'Oh dear! Oh dear! I shall be late!' (when she thought it over afterwards, it occurred to her that she ought to have wondered at this, but at the time it all seemed quite natural); but when the Rabbit actually *took a watch out of its waistcoat-pocket*, and looked at it, and then hurried on, Alice started to her feet, for it flashed across her mind that she had never before seen a rabbit with either a waistcoat-pocket, or a watch to take out of it, and burning with curiosity, she ran across the field after it, and fortunately was just in time to see it pop down a large rabbit-hole under the hedge.

In another moment down went Alice after it, never once considering how in the world she was to get out again.

Sõnastikud - motivatsioon

- Lihtsustus: ei tee vahet suurtel ja väikestel tähtedel
- Võimalikud lahendused:

Sõnastikud - motivatsioon

- Lihtsustus: ei tee vahet suurtel ja väikestel tähtedel
- Võimalikud lahendused:

  1. Luua 26 loendurit. Käia tekst sümbolhaaval läbi ja tähe esinemisel suurendada loendurit.

Sõnastikud - motivatsioon

- Lihtsustus: ei tee vahet suurtel ja väikestel tähtedel
- Võimalikud lahendused:

  1. Luua 26 loendurit. Käia tekst sümbolhaaval läbi ja tähe esinemisel suurendada loendurit.

  2. Luua list, kus on 26 elementi. Käia tekst sümbolhaaval läbi ja tähe esinemisel suurendada listi vastavat elementi.

Sõnastikud - motivatsioon

- Lihtsustus: ei tee vahet suurtel ja väikestel tähtedel
- Võimalikud lahendused:

  1. Luua 26 loendurit. Käia tekst sümbolhaaval läbi ja tähe esinemisel suurendada loendurit.

  2. Luua list, kus on 26 elementi. Käia tekst sümbolhaaval läbi ja tähe esinemisel suurendada listi vastavat elementi.

```
tekst = 'abrakadabra'
loendur = [0]*26
for el in tekst:
    i = ord(el.lower()) - 97
    loendur[i] += 1
print(loendur)
```

Sõnastikud - motivatsioon

- Lihtsustus: ei tee vahet suurtel ja väikestel tähtedel
- Võimalikud lahendused:

  1. Luua 26 loendurit. Käia tekst sümbolhaaval läbi ja tähe esinemisel suurendada loendurit.

  2. Luua list, kus on 26 elementi. Käia tekst sümbolhaaval läbi ja tähe esinemisel suurendada listi vastavat elementi.

```
tekst = 'abrakadabra'
loendur = [0]*26
for el in tekst:
    i = ord(el.lower()) - 97
    loendur[i] += 1
print(loendur)
```

```
[5, 2, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0
, 0, 0, 0, 0, 0]
```

Sõnastikud - motivatsioon

- Tulemuste selgem esitamine

Sõnastikud - motivatsioon

- Tulemuste selgem esitamine

```
tekst = 'abrakadabra'
loendur = [0]*26
for el in tekst:
    i = ord(el.lower()) - 97
    loendur[i] += 1
for i in range(len(loendur)):
    if loendur[i] != 0:
        print(str(chr(i + 97)) + ': ' + str(loendur[i]))
```

Sõnastikud - motivatsioon

- Tulemuste selgem esitamine

```
tekst = 'abrakadabra'
loendur = [0]*26
for el in tekst:
    i = ord(el.lower()) - 97
    loendur[i] += 1
for i in range(len(loendur)):
    if loendur[i] != 0:
        print(str(chr(i + 97)) + ': ' + str(loendur[i]))
```

```
a: 5
b: 2
d: 1
k: 1
r: 2
```

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')]
```

List vs sõnastik

```
>>> t_lst=list()
>>>
>>> t_lst.append(15)
>>>
>>> t_lst.append(20)
>>>
>>> t_lst.append(39)
>>>
>>> print(t_lst)
[15, 20, 39]
>>>
>>> t_lst[2] = 40
>>>
>>> print(t_lst)
[15, 20, 40]
```

List vs sõnastik

```
>>> t_lst=list()
>>>
>>> t_lst.append(15)
>>>
>>> t_lst.append(20)
>>>
>>> t_lst.append(39)
>>>
>>> print(t_lst)
[15, 20, 39]
>>>
>>> t_lst[2] = 40
>>>
>>> print(t_lst)
[15, 20, 40]
```

```
>>> t_dd=dict()
>>>
>>> t_dd['vahe1']=15
>>>
>>> t_dd['vahe2']=20
>>>
>>> t_dd['loputest']=39
>>>
>>> print(t_dd)
{'loputest': 39, 'vahe2': 20, 'vahe1': 15}
>>>
>>> t_dd['loputest']=40
>>>
>>> print(t_dd)
{'loputest': 40, 'vahe2': 20, 'vahe1': 15}
>>>
```

Sõnastikud

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

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

- 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.

Näide

```
phonebook = {'Alice':   '2341',
             'Betti':   '9102',
             'Anu':     '3158',
             'Eliise':  '0142',
             'Annely':  '5551'}
nr = phonebook['Eliise']
print(nr)
```

```
0142
```

Sõnastikud

- Üle sõnastiku itereerimiseks saab kasutada for-tsüklit

Näide

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

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

```
Python    Guido van Rossum
Tcl       John Ousterhout
Perl      Larry Wall
```

Sõnastikud

Eeldefineeritud funktsioone ja meetodeid

Meetod	Kirjeldus
k **in** D	kas võti k on sõnastikus?
D.keys()	tagastab võtmete dünaamilise vaate
D.values()	tagastab väärtuste dünaamilise vaate
D.items()	tagastab (võti,väärtus)-paaride dünaamilise vaate
D.copy()	tagastab sõnastiku koopia
D.pop(k)	tagastab indeksiga k väärtuse ja eemaldab selle sõnastikust
del D[k]	eemaldab indeksiga k väärtuse sõnastikust
len(D)	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}
```

Sõnastikud - esialgne ülesanne

- Ülesanne
  On antud tekst. Lugeda kokku, mitu korda iga täht selles esineb.

Sõnastikud - esialgne ülesanne

- Ülesanne
  On antud tekst. Lugeda kokku, mitu korda iga täht selles esineb.

```
tekst = 'abrakadabra'
sõnastik = {}
for el in tekst:
    el = el.lower()
    if el in sõnastik:
        sõnastik[el] += 1
    else:
        sõnastik[el] = 1
for k,v in sõnastik.items():
    print(k + ': ' + str(v))
```

Sõnastikud - esialgne ülesanne

- Ülesanne
  On antud tekst. Lugeda kokku, mitu korda iga täht selles esineb.

```
tekst = 'abrakadabra'
sõnastik = {}
for el in tekst:
    el = el.lower()
    if el in sõnastik:
        sõnastik[el] += 1
    else:
        sõnastik[el] = 1
for k,v in sõnastik.items():
    print(k + ': ' + str(v))
```

```
k: 1
b: 2
r: 2
a: 5
d: 1
```

Sõnastikud - esialgne ülesanne

- Ülesanne
  On antud tekst. Lugeda kokku, mitu korda iga täht selles esineb.

Sõnastikud - esialgne ülesanne

- Ülesanne
  On antud tekst. Lugeda kokku, mitu korda iga täht selles esineb.

```
tekst = 'abrakadabra'
sõnastik = {}
for el in tekst:
    el = el.lower()
    sõnastik[el] = sõnastik.get(el, 0) + 1
for k, v in sõnastik.items():
    print(k + ': ' + str(v))
```

Sõnastikud - esialgne ülesanne

- Ülesanne
  On antud tekst. Lugeda kokku, mitu korda iga täht selles esineb.

```
tekst = 'abrakadabra'
sõnastik = {}
for el in tekst:
    el = el.lower()
    sõnastik[el] = sõnastik.get(el, 0) + 1
for k, v in sõnastik.items():
    print(k + ': ' + str(v))
```

```
d: 1
b: 2
a: 5
k: 1
r: 2
```

Sõnastikud - esialgne ülesanne

- Ülesanne
  On antud raamat 'Alice in Wonderland'. Lugeda kokku, mitu korda iga täht selles esineb.

Sõnastikud - esialgne ülesanne

- Ülesanne
  On antud raamat 'Alice in Wonderland'. Lugeda kokku, mitu korda iga täht selles esineb.

```
f = open('alice.txt')
tekst = f.read()
f.close()
sõnastik = {}
for el in tekst:
    el = el.lower()
    if el.isalpha():
        sõnastik[el] = sõnastik.get(el, 0) + 1
for k, v in sõnastik.items():
    print(k + ': ' + str(v), end = ' ')
```

Sõnastikud - esialgne ülesanne

- Ülesanne
  On antud raamat 'Alice in Wonderland'. Lugeda kokku, mitu korda iga täht selles esineb.

```
f = open('alice.txt')
tekst = f.read()
f.close()
sõnastik = {}
for el in tekst:
    el = el.lower()
    if el.isalpha():
        sõnastik[el] = sõnastik.get(el, 0) + 1
for k, v in sõnastik.items():
    print(k + ': ' + str(v), end = ' ')
```

```
o: 8145 c: 2399 l: 4716 m: 2107 j: 146 d: 4931 p: 1524 y: 2263
n: 7015 f: 2001 t: 10689 x: 148 k: 1158 e: 13576 v: 846 h: 7375
z: 78 s: 6501 b: 1476 w: 2676 q: 209 u: 3468 i: 7515 a: 8791 g:
2531 r: 5439
```

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 = ' ')

# Trükib:
r  d  b  c  a
```

Tehted hulkadega

Matemaatiline märk	*Pythoni märk*	*Kirjeldus*
$\in$	`in`	on hulga element
$\notin$	`not in`	ei ole hulga element
$=$	`==`	on võrdne
$\neq$	`!=`	mittevõrdne
$\subset$	`<`	on range alamhulk
$\subseteq$	`<=`	on alamhulk
$\supset$	`>`	on range ülemhulk
$\cap$	`&`	ühisosa
$\cup$	`|`	ühend
$-$	`-`	vahe
Δ	`^`	sümmeetriline 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)              # Trükib: True
print(2 not in s1)          # Trükib: False
print(s1 == s2)             # Trükib: False
print(s2 < s1)              # Trükib: True
print(s1 & s2)              # Trükib: {2, 3}
print(s1 | s3)              # Trükib: {0, 1, 2, 3, 4, 6, 7}
print(s1 ^ s4)              # Trükib: {0, 1, 4, 5, -1}
```

Hulgad

Eeldefineeritud funktsioone ja meetodeid

Meetod	Kirjeldus
S.add(el)	lisab elemendi *el* hulka *S* (juhul kui seda varem polnud)
S.update(S1)	täiendab hulka *S* teise hulga *S1* elementidega
S.remove(el)	eemaldab elemendi *el* hulgast *S*
S.pop()	eemaldab ja tagastab hulgast *S* suvalise elemendi
S.clear()	tühjendab hulga *S*
S.copy()	tagastab hulga *S* koopia (shallow)

Ülesanne

- Koostada funktsioon, mis tagastab tõeväärtuse, kas etteantud listis pikkusega n sisalduvad kõik naturaalarvud 1...n

Ülesanne

- Koostada funktsioon, mis tagastab tõeväärtuse, kas etteantud listis pikkusega n sisalduvad kõik naturaalarvud 1...n

```
def naturaal(x):
  y = []
  for i in range(1, len(x) + 1):
    y.append(i)
  return set(x) == set(y)

list1 = [1, 3, 6, 4, 2, 5, 8, 9, 7]
print(naturaal(list1))
```

Ülesanne

- Koostada funktsioon, mis tagastab tõeväärtuse, kas etteantud listis pikkusega n sisalduvad kõik naturaalarvud 1...n

```
def naturaal(x):
  y = []
  for i in range(1, len(x) + 1):
    y.append(i)
  return set(x) == set(y)

list1 = [1, 3, 6, 4, 2, 5, 8, 9, 7]
print(naturaal(list1))
```

```
True
```

Suur tänu osalemast

ja

kohtumiseni!