

PROG1: Prednáška 9

Zoznamy (lists)

Cast prva: Zaklady

Projekt

O projekte:

- Je za 20 bodov.
- Termin odovzdanja: 12.12. 23:59
- Je tazsi ako hocico, co sme doteraz robili!
- Zacnite ho riesit co najskor!
- Budete potrebovat ovladat pracu so zoznamami!
(zoznamy budeme preberat tento a este aj
buduci tyzden)

Kratkodoby plan

T9 (aktualny tyzden):

- Prednaska: Zoznamy (zaklady)
- Cvicenia: ulohy na zoznamy

T10:

- Prednaska: Zoznamy (pokrocilejsie casti)
- Cvicenia: ulohy na zoznamy

T11:

- Prednaska: preriesim ulohy z desiateho tyzdna, nebudem preberat novu latku
- Cvicenia: posledna sanca konzultovat s cviciacim pred druhym testom

T12:

- Prednaska:
 - druhy test
 - zaverecne informacie
- Cvicenia: posledna sanca konzultovat s cviciacim projekt

Domaca uloha

1. Precitajte si sekcie 10.1 – 10.9 v knihe.
2. Vyrieste ulohy z Cvicenia 9.

Zoznamy

Zoznam je postupnosť hodnôt.

Poznámka:

Retazec je tiež postupnosť. V prípade retazca musia byť prvkami postupnosti znaky. V prípade zoznamu môžu byť prvky ľubovoľného typu.

Vytvaranie zoznamov

Na vytvorenie zoznamu pouzivame hranate zatvorky.

Priklady:

```
t1=[] #prazdny zoznam
```

```
t2=['a', 'mama', 2, 2.5, [1,2,3]]
```

```
t3=[2,2,2,2,2]
```

Pristupovanie k prvkom v zozname

Rovnako ako pri retazcoch.

Priklad:

```
>>> t=['a', 'mama', 2, 2.5, [1,2,3]]
>>> t[1]
'mama'
```

Pozor! Rovnako ako v retazcoch su prvky v zozname indexovane od **nuly!**

Ak chceme ziskat prvky prvok zoznamu t , musime zadat $t[0]$. ($t[1]$ vrati druhy prvok zoznamu)

Zoznam je menitelny

Narozdiel od retazcov, zoznamy su menitelne.

Priklad:

```
>>> t=[42,123]
>>> t[1]=5
>>> t
[42, 5]
```


Operator in

Operator in funguje aj pre zoznamy.

Priklad:

```
>>> t=[42,123]
>>> 42 in t
True
```

Dĺzka zoznamu

Dĺzku zoznamu dostaneme pomocou funkcie **len()** (rovnako ako pri retazcoch).

Priklad:

```
>>> t=['a', 'mama', 2, 2.5, [1,2,3]]  
>>> len(t)  
5
```

Zoznam t obsahuje 5 prvkov: jeden znak, jeden retazec, jeden integer, jeden float, a jeden zoznam.

Prechadzanie cez zoznam

Prvy sposob: prechadzanie cez prvky zoznamu

Priklad:

```
>>> t=['a', 2, [1,2,3]]
>>> for item in t:
        print(item)

a
2
[1, 2, 3]
>>> |
```

Prechadzanie cez zoznam

Druhý spôsob: prechadzanie cez indexy zoznamu

Priklad:

```
>>> t=[1,2,3]
>>> for i in range(len(t)):
        t[i]=t[i]*2
```

```
>>> t
[2, 4, 6]
```

Tento spôsob je prirodzenejší, ak chceme prepisovať prvky zoznamu.

Operacie so zoznamami: operator +

```
>>> t1=[7,8,9]
>>> t2=[1,2,3]
>>> t=t1+t2
>>> t
[7, 8, 9, 1, 2, 3]
```

Operacie so zoznamami: operator *

```
>>> t1=[1,2,3]
>>> t2=t1*2
>>> t2
[1, 2, 3, 1, 2, 3]
```

Vyseky zo zoznamu

```
>>> t=['a','b','c','d','e','f']  
>>> t[1:3]  
['b', 'c']
```

```
>>> t=['a','b','c','d','e','f']  
>>> t[1:3]=['x','y']  
>>> t  
['a', 'x', 'y', 'd', 'e', 'f']
```

Metody pre zoznamy

Aj pre zoznamy mame vela metod.

Velmi uzitocna je metoda **append**.

```
>>> t=['a','b','c']
>>> t.append('d')
>>> t
['a', 'b', 'c', 'd']
```

Pozor! Porovnajte prikazy vyssie s prikazmi:

```
>>> t=['a','b','c']
>>> t2=t.append('d')
>>> t2
>>>
```


Metody pre zoznamy

Pri retazcoch vacsina metod **vraťi** zmeneny retazec. Takze z retazca `ret` mozeme vytvorit retazec `ret2` napisany velkymi pismenami pomocou prikazu:
ret2=ret.upper()

Pri zoznamoch vacsina metod zmeni zoznam, ale **vraťi** hodnotu **None**. Takze, ak zadame prikaz
t2=t.append('d')
zoznam `t` sa zmeni, ale do `t2` sa ulozi iba hodnota `None`.

POINTA: Treba citat dokumentaciu k metodam!

Vymazavanie zo zoznamu

Metoda pop:

```
>>> t = ['a', 'b', 'c']
>>> x = t.pop(1)
>>> t
['a', 'c']
>>> x
'b'
```

Prikaz `t.pop(1)` vymaze zo zoznamu hodnotu na indexe 1 a vrati tuto hodnotu (v priklade vyssie sme si tuto hodnotu ulozili do premennej `x`).

Vymazavanie zo zoznamu

Operator del:

```
>>> t = ['a', 'b', 'c']
```

```
>>> del t[1]
```

```
>>> t
```

```
['a', 'c']
```

Vymazavanie zo zoznamu

Metoda remove:

```
>>> t=['a', 'b', 'c', 'b']
>>> t.remove('b')
>>> t
['a', 'c', 'b']
```

Pozor! Prikaz `t.remove('b')` vymaze iba prve 'b' v zozname.

Narozdiel od metody `pop` a operatora `del`, zadavame pri metode `remove` element, ktory chceme vymazat (nie jeho index).

Zoznamy a retazce

Retazec nie je to iste ako zoznam znakov! (to by malo byt jasne, lebo sme si povedali, ze retazce su nemenne a zoznamy su menitelne)

Ak chceme z retazca urobit zoznam znakov, mozeme pouzit funkciu `list`.

```
>>> ret='mama'  
>>> ret  
'mama'  
>>> t=list(ret)  
>>> t  
['m', 'a', 'm', 'a']
```

Zoznamy a retazce: metoda split

Pomocou metody split, mozeme rozdelit retazec na zoznam retazcov.

```
>>> ret='Monty Python'  
>>> t=ret.split()  
>>> t  
['Monty', 'Python']
```

Do metody split mozeme ako argument zadat znak, podla ktoreho, chceme retazec rozdelovat. Ak ziadny znak nezadame, rozdeluje sa podla medzier.

```
>>> ret='LA-LA-LA'  
>>> t=ret.split('-')  
>>> t  
['LA', 'LA', 'LA']
```