Programování 1: Seznamy, řezy a řetězce

Petr Čermák
cermak@mag.mff.cuni.cz

Katedra Fyziky Kondenzovaných Látek
MFF UK Praha

2020
forked from https://gitlab.kam.mff.cuni.cz/mj/prm1

List Slicing / Řezy seznamem

```
>>> x = [11, 22, 33, 44, 55, 66, 77]
>>> x[2:5]
[33, 44, 55]
```
(podseznam)

```
>>> x[:3]
[11, 22, 33]
```
(prefix)

```
>>> x[5:]
[66, 77]
```
(suffix)

```
>>> x[:]
```
(x.copy() je univerzálnější)

```
[11, 22, 33, 44, 55, 66, 77]
```
(kopie seznamu)

```
>>> x[::2]
[11, 33, 55, 77]
```
(každý druhý prvek)

```
>>> x[::-1]
[77, 66, 55, 44, 33, 22, 11]
```
(pozpátku)

Operace se seznamy

```
>>> 3 in [1, 2, 3, 4, 5]
True
```
(projde seznam a zjistí, zda v něm je daný prvek)

```
>>> x = [1, 2, 3, 4, 5]
>>> x.pop()
5
```
(odebere z konce)

```
>>> x.pop(0)
1
```
(odebere na zadané pozici)

```
>>> x
[2, 3, 4]
```
(co zbylo)

```
>>> x.insert(1, 42)
>>> x
[2, 42, 3, 4]
```
(vloží na zadanou pozici)

```
>>> x.extend([16, 20])
>>> x
[2, 42, 3, 4, 16, 20]
```
(přidá iterable)

Generátorové notace / List comprehension

Je to rychlé a přehledné

```
>>> a = [x*x for x in range(10)]
>>> a
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

```
>>> a = [x for x in range(50) if x % 7 == 0]
>>> a
[0, 7, 14, 21, 28, 35, 42, 49]
```

Lze i pro sety (unikátní list):

```
>>> a = {x for x in range(50) if x % 7 == 0}
>>> a
{0, 7, 14, 21, 28, 35, 42, 49}
```

Fronta s dvěma konci

Podobné jako list, ale umí rychle přidávat a odebírat ze začátku.

```
1  from collections import deque
2
3  N = 99999
4  a = [x for x in range(N)]
5  b = deque(a)
6
7  # tohle je rychlé (čas 1)
8  a.pop()
9  b.pop()
10
11 a.pop(0) # tohle ale trvá N času
12
13 b.popleft() # deque je rychlý
14
15 # obdobně s vkládáním
16 b.appendleft(-1)
```

Časování v notebooku

```
1 def nadruhou(n):
2     return n*n
```

```
>>> %timeit nadruhou(10)
132 ns ± 4.96 ns per loop (mean ± std. dev. of 7
runs, 10000000 loops each)

>>> %timeit -r1 -n10 nadruhou(10)
290 ns ± 0 ns per loop (mean ± std. dev. of 1 run,
10 loops each)

>>> %%timeit -r1 -n10    (celá buňka, 2x %)
>>> nadruhou(10)
>>> nadruhou(10)
360 ns ± 0 ns per loop (mean ± std. dev. of 1 run,
10 loops each)
```

Řetězce se chovají jako seznamy

```
>>> x="Sedmikráska"
>>> len(x)
11

>>> x[0]
'S'

>>> x[:4] + x[10]
'Sedma'

>>> x[0]="s"
<chyba>    (řetězce nelze měnit)

>>> for a in x:
...     print(a)
S
e
...
```

Operace s řetězci

```
>>> "velká".upper()
'VELKÁ'

>>> "banana".find("na")
2

>>> "banana".find("baba")
-1

>>> "Na počátku bylo   slovo.".split()
['Na', 'počátku', 'bylo', 'slovo.']

>>> "+".join(["Alice", "Bob", "Cyril"])
'Alice+Bob+Cyril'

>>> "1+2+3".split(sep="+")
['1', '2', '3']

>>> [p + q for p in '123' for q in 'ab']
['1a', '1b', '2a', '2b', '3a', '3b']
```

Lexikografické porovnávání

```
>>> [1, 2, 3] < [1, 2, 5]
True

>>> [1, 2, 3] < [1, 2, 3, 4]
True

>>> [1, 2, 3] == [1, 2, 3]
True

>>> "červ" > "čert"
True

>>> "červ" > "datel"
True
```

(pozor, nerespektuje česká pravidla)

Převody mezi typy dat

```
>>> int("123")
123

>>> str(123)
"123"

>>> list("123")
['1', '2', '3']

>>> str(['1', '2', '3'])
"['1', '2', '3']"

>>> list(range(1, 10))
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Úkoly na hodinu

otočte řetězec (datel $\rightarrow$ letad)

otočte číslo v desítkové soustavě (1024 $\rightarrow$ 4201)

spočítejte, kolik zadaný řetězec obsahuje různých slov

vyhodnoťte výraz se sčítáním (12+34+1 $\rightarrow$ 47)