Programování 1.7: Dictionaries/Slovníky

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

Slovníky

```
>>> teploty = { "Praha": 17, "Dillí": 42,
"Longyearbyen": -46 }
>>> teploty["Praha"]
17

>>> teploty["Horní Dolní"] = 11
>>> del teploty["Horní Dolní"]
>>> "Horní Dolní" in teploty
False

>>> teploty["Peklo"]
<chyba KeyError>

>>> teploty.get("Peklo")
None

>>> teploty.get("Peklo", -999)
-999
```

(i pro pole)

Cyklus přes prvky slovníku

```
>>> for k in teploty.keys():
...     print(k)
Praha
Dillí
Longyearbyen
```

(případně lze keys vynechat)

```
>>> for v in teploty.values():
...     print(v)
17
42
-46
```

```
>>> for k, v in teploty.items():
...     print(f"{k} = {v}")
Praha = 17
Dillí = 42
Longyearbyen = -46
```

List comprehension pro množiny a slovníky

```
>>> [ x**2 for x in range(5) ]
[0, 1, 4, 9, 16]
```
(list)

```
>>> { x**2 for x in range(5) }
{0, 1, 4, 9, 16}
```
(set)

```
>>> { x: x**2 for x in range(5) }
{0: 0, 1: 1, 2: 4, 3: 9, 4: 16}
```
(dict)

Pár triků

Funkce s více výsledky pomocí tuple

```
1 def deleni(x, y):
2     return x//y, x%y
3
4 podil, zbytek = deleni(1000, 7)
```

Prázdné sety vs. slovníky

```
>>> type({ 1 })
<class 'set'>

>>> type({})
<class 'dict'>

>>> type(set())
<class 'set'>
```

Příklad: Zipování seznamů

```
>>> x = [1, 2, 3]
>>> y = ["a", "b", "c"]
>>> list(zip(x, y))
[(1, 'a'), (2, 'b'), (3, 'c')]

>>> for i, j in zip(x, y):
>>>     print(i, j)
1 a
2 b
3 c
```

List comprehension: Další triky

```
>>> [int(s) for s in input().split()]
>>> 1 2 3 4 5
[1, 2, 3, 4, 5]
```

(načtení seznamu čísel na jednom řádku)

```
>>> s=[[1, 2, 3], [4, 5]]
>>> [x for t in s for x in t]
[1, 2, 3, 4, 5]
```

(„zploštění“ seznamu)

```
>>> m=[[0]*5 for _ in range(3)]
>>> m
```

(výroba matice: seznam 3 různých seznamů nul)

```
[[0,0,0,0,0], [0,0,0,0,0], [0,0,0,0,0]]
```

```
>>> n=[[1, 2, 3], [4, 5, 6]]
>>> [[r[i] for r in n] for i in range(3)]
[[1,4], [2,5], [3,6]]
```

(transpozice matice)

Agregační funkce

```
>>> s = [2**x % 37 for x in range(10)]
>>> s
[1, 2, 4, 8, 16, 32, 27, 17, 34, 31]
>>> min(s)
1
>>> max(s)
34
>>> sum(s)
172
>>> all([x < 37 for x in s])
True
```
(and přes všechny prvky seznamu)

```
>>> any([x > 33 for x in s])
True
```
(or přes všechny prvky seznamu)

```
>>> sorted(s)
[1, 2, 4, 8, 16, 17, 27, 31, 32, 34]
```

Závěrem

- Klíčem může být libovolný immutable typ (třeba číslo, řetězec či tuple, ale ne seznam nebo množina).
- Operace s jednotlivými prvky běží v konstantním čase (aspoň průměrně).
- Operace s celou množinou běží v lineárním čase.
- Shrnutí:
  - Tuple ... static array
  - List ... dynamical array
  - Dictionary ... hash table
  - Set ... Dictionary bez hodnot

Úkoly na hodinu

- vytvoří tabulku násobilky (a×b pro všechna a, b od 1 do daného čísla)
- zjistí průnik dvou (neuspořádaných) seznamů
- vybere z textu slova, která jsou palindromická (např. kajak)
- spočítá skalární součin dvou vektorů
- seřadí slova na řádku podle jejich délky (hint: (1) slova převést na dvojice (délka, slovo), (2) setřídit, (3) vypsat slova)

```
1  def nasobilka(n):
2
3  def prunik_seznamu(x, y):
4
5  def palindromy(text):
6
7  def skalarni_soucin(x, y):
8
9  def slova_podle_delky(radek):
```