IB113 Úvod do programování a algoritmizace

Cvičení 9

Jaromír Plhák

Kódy z minula (řadící algoritmy)
Vyhodnocení úkolu 3

Zadání domácí úlohy 4 – Padající piškvorky

- V ISu v odevzdávárně
  - https://is.muni.cz/auth/el/1433/podzim2017/IB113/ode/72006396/72006443/hw04_zadani.py (skupina 1)
  - https://is.muni.cz/auth/el/1433/podzim2017/IB113/ode/72006400/72006457/hw04_zadani.py (skupina 2)
- Veškeré informace v souboru
- Odevzdejte pouze tento soubor do odevzdávárny
  - https://is.muni.cz/auth/el/1433/podzim2017/IB113/ode/72006396/72006432/ (skupina 1)
  - https://is.muni.cz/auth/el/1433/podzim2017/IB113/ode/72006400/72006453/ (skupina 2)
- Do **čtvrtka** 23. 11. 2017, 23:59

Datové typy – n-tice

- Definice: (item1, item2, . . .)
- Přesne jako seznam, ale nesmí být změněná
  - Nesmíme přidávat, modifikovat ani mazat prvky
  - Rychlejší, bezpečnější proti náhodným chybám

```
>>> myList = [1, 2, 3]
>>> myList.append(4)
>>> myTuple = (1, 2, 3)
>>> myTuple.append(4)
Traceback (most recent call last):
File "C:/Python32/test", line 9, in
myTuple.append(4)
AttributeError: 'tuple' object has no
    attribute 'append'
```

Datové typy - množina

- Definice: set(item1, item2, . . .) nebo {item1, item2, . . . }
- Přesně jako seznam, ale
  - Bez duplicitních položek
  - Nezachovává pořadí prvků!

```
>>> myList = "my name is Eric and Eric is
    my name".split()
>>> myList
['my', 'name', 'is', 'Eric', 'and', 'Eric
   ', 'is', 'my', 'name']
>>> mySet = set(myList)
>>> mySet
{'name', 'is', 'Eric', 'and', 'my'}
```

Datové typy - množina

- Množina je měnitelná (na rozdíl od n-tice)
  - Metody add(), remove(), pop(), discard()
    - discard() jen pokud klíč existuje – nevyhazuje chybu
  - Neměnitelná množina: frozenset()

```
>>> cities = set(["Brno", "Praha"])
>>> cities.add("Ostrava")
>>> cities
set(['Ostrava', 'Praha', 'Brno'])
```

Datové typy – funkce nad množinami

- Klasické množinové operace: průnik, sjednocení, rozdíl, symetrický rozdíl
- Test na podmnožinu/nadmnožinu
- Nech a, b jsou množiny:

a.intersection(b) # Prunik

a.union(b) # Sjednoceni

a.difference(b) # Rozdil

a.symmetric_difference(b) # Sym. rozdil

a.issubset(b) # Podmnozina

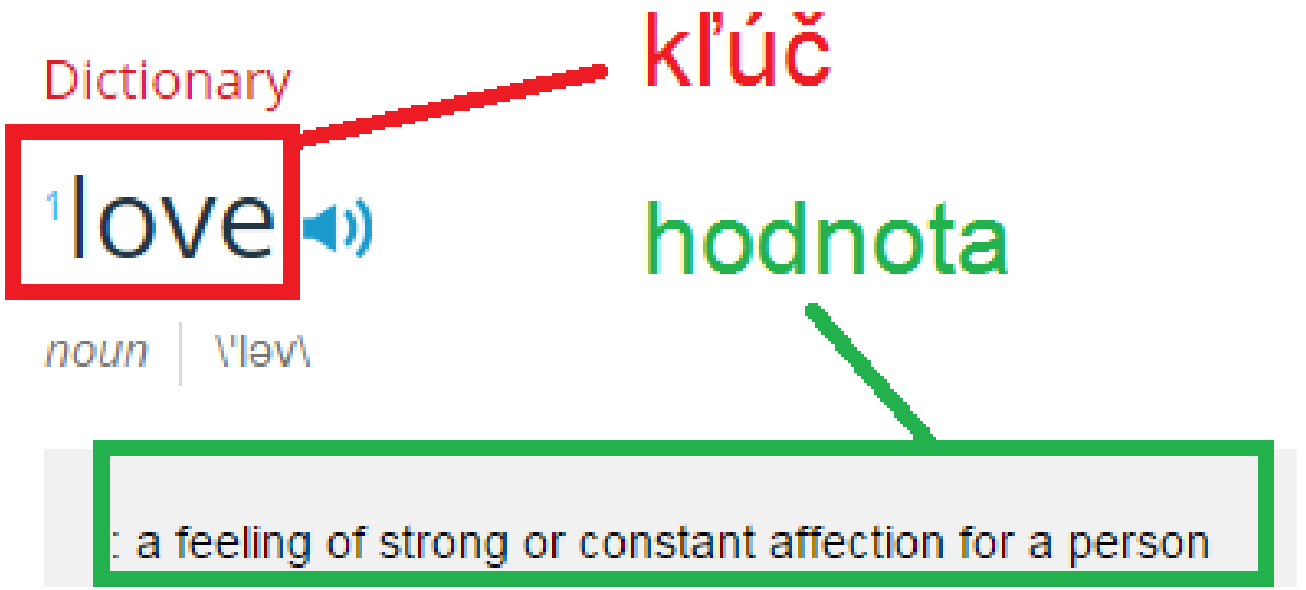
a.issuperset(b) # Nadmnozina

Úkoly

- 9.4.1. Kontrola unikátnosti seznamu
  - Napište funkci, která zkontroluje, zda předaný seznam obsahuje jen unikátní položky.
- 9.4.2. Kontrola řádku Sudoku
  - Napište funkci, která zkontroluje, zda předaný seznam reprezentuje správný řádek vyplněného Sudoku, tj. obsahuje pouze čísla 1 až 9 a každé z nich právě jedenkrát.
- Použijte množiny!

Datové typy - slovník

- Kolekce párů (klíč – hodnota)
- Každý **klíč** je jedinečný (vyskytuje se ve slovníku jen jednou)
- **Hodnoty** se ve slovníku mohou opakovat



Zdroj: http://www.merriam-webster.com/dictionary/love

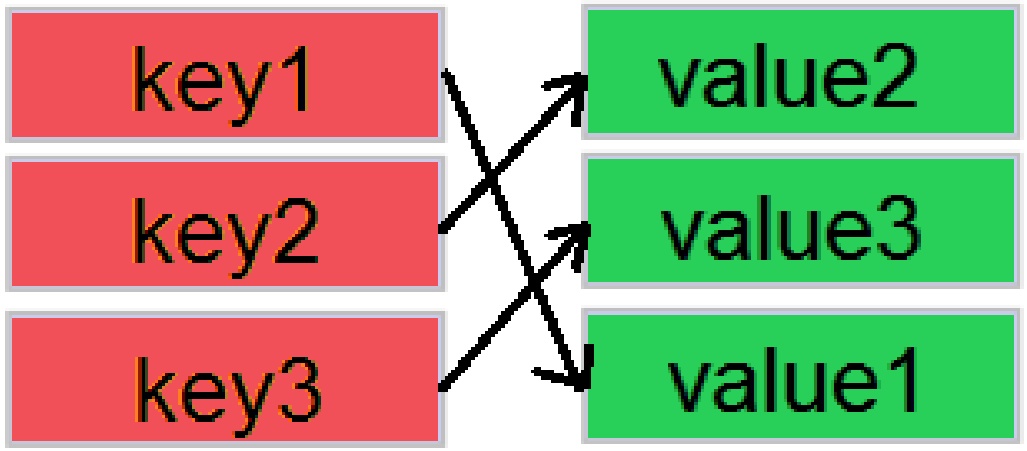
Datové typy - slovník

- Slovník v Pythonu podporuje tyto operace:
  - d = {} – vytvoření prázdného slovníku
  - d[k] = v – přidaní klíče k do slovníku a jeho svázání s hodnotou v
  - d[k] = v – svázání existujícího klíče k s jinou hodnotou v
  - del d[k], d.pop(k) – odebrání klíče k (a zároveň i odpovídající hodnoty) ze slovníku
  - d[k] – nalezení hodnoty, která je svázaná s klíčem k
  - len(d) – počet prvků ve slovníku

Datové typy – slovník v Pythonu

- Definice:

```
my_dictionary = {
        'key1': 'value1',
        'key2': 'value2',
        'key3': 'value3‘
}
```



- Klíč je tak trochu jako index v seznamu
  - Zpřístupňujeme přes něho danou hodnotu
  - Ale není uspořádaný

Datové typy – příklad slovníku

- Výkladový slovník Merriam-Webster
  - http://www.merriam-webster.com/

```
definitions = {
    "love"   : "a feeling of strong or
       constant affection for a person",
    "friend" : "a person who you like and
        enjoy being with",
    "truth"  : "the property (as of a
       statement) of being in accord with
        fact or reality"
}
```

Datové typy – příklad slovníku

```
phonebook = {
    "Peter" : 147258,
    "Jane" : 789456,
    "Mary" : 333333
}
```

```
phonebook = {}
phonebook["Peter"] = 147258
phonebook["Jane"] = 789456
phonebook["Mary"] = 333333
```

```
phonebook = dict(Peter=147258, Jane
   =789456, Mary=333333)
```

Datové typy – příklad slovníku

- Osoba a její oblíbené ovoce

```
fruits = {
        "Peter" : "apple",
        "Jane" : "banana",
        "Mary" : "apple"
}
```

- Hodnoty se mohou opakovat!
- Ale klíče ne – musí být jednoznačné (jinak bychom nevěděli, jakou osobu myslíme)
- Další příklady
  - http://www.python-course.eu/python3_dictionaries.php

Datové typy – operace se slovníkem

- Vytvoření prázdného slovníku:
  - **d = {}**
- Přidání klíče k do slovníku a jeho svázání s hodnotou v:

```
>>> d["Peter"] = "apple"
>>> d["Jane"] = "banana"
>>> d["Mary"] = "apple"
>>> d
{'Jane': 'banana', 'Peter': 'apple', '
    Mary': 'apple'}
```

- Pořadí není zachované!
  - Proto nemůžeme používat uspořádané indexy

Datové typy – operace se slovníkem

```
>>> d
{'Jane': 'banana', 'Peter': 'apple', '
    Mary': 'apple'}
```

- Svázání existujícího klíče k s jinou hodnotou v:

```
>>> d["Mary"] = "coconut"
>>> d
{'Jane': 'banana', 'Peter': 'apple', '
    Mary': 'coconut'}
```

- Slovník je měnitelný (jako seznam)
- Víc stejných klíčů neexistuje – jeden se přepisuje

Datové typy – operace se slovníkem

```
>>> d
{'Jane': 'banana', 'Peter': 'apple', '
    Mary': 'coconut'}
```

- Odebrání klíče k (a zároveň i odpovídající hodnoty) ze slovníku:

```
>>> del d["Mary"] # Option 1
>>> d.pop("Peter") # Option 2
'apple'
>>> d
{'Jane': 'banana'}
```

Datové typy – operace se slovníkem

```
>>> d
{'Jane': 'banana'}
```

- Nalezení hodnoty, která je svázaná s klíčem

```
>>> d["Jane"]
'banana'
```

- Počet prvků ve slovníku

```
>>> len(d)
1
```

Datové typy – operace se slovníkem

- Syntaxe: dictionary.function()
- keys() – všechny **klíče** slovníku
- values() – všechny **hodnoty** slovníku
- items() – všechny **dvojice** (klíč, hodnota)

```
>>> d = {"Peter" : "apple", "Jane" : "
    banana", "Mary" : "apple"}
>>> for key in d.keys():
        print(key, d[key])

Jane banana
Peter apple
Mary apple
```

Úkoly

- 9.3.2. Frekvenční analýza písmen
  - Napište funkci freq_analysis(text), která spočítá výskyt jednotlivých písmen (znaků) ve vstupním textu a následně tento seznam vypíše setříděný sestupně podle počtu výskytů.
- 9.3.3. Frekvenční analýza slov
  - Napište funkci freq_analysis(text), která spočítá výskyt jednotlivých slov ve vstupním textu a následně tento seznam vypíše setříděný podle abecedy.
  - Tip: může se vám hodit metoda split, která je použitelná na řetězcích
    - "a b c".split()