České vysoké učení technické v Praze FIT

Programování v Pythonu

Jiří Znamenáček

Příprava studijního programu Informatika je podporována projektem financovaným z Evropského sociálního fondu a rozpočtu hlavního města Prahy.

Praha & EU: Investujeme do vaší budoucnosti

Slovníky

1. Zanalyzujte vstupní textový soubor následujícím způsobem: Vytvořte slovník, jehož klíči budou jednotlivá písmena a hodnotami počet jejich výskytů v textu. Uvedený slovník vypište pomocí `pprint.pprint()`.

[-] řešení (*typy/slovniky/count_letters.1.py*)

```
import pprint

text = ''
with open('leacock-abc.txt', mode='r', encoding='utf-8') as
f:
    text = f.read()

statistika = {}
for znak in text:
    # přeskoč nepísmenné znaky
    if not znak.isalpha():
        continue
    # písmena zahrň do slovníku
    if znak not in statistika:
        statistika[znak] = 1
    else:
        statistika[znak] += 1

pprint.pprint(statistika)
```

2. Rozšiřte předchozí program tak, že hodnotou bude n-tice o dvou prvcích – prvním bude počet výskytů daného písmene v textu, druhým jeho relativní četnost v % (vzhledem k počtu všech písmen v textu).

[-] řešení (*typy/slovniky/count_letters.2.py*)

```
import pprint

text = ''
with open('leacock-abc.txt', mode='r', encoding='utf-8') as
f:
    text = f.read()

statistika = {}
for znak in text:
    # přeskoč nepísmenné znaky
    if not znak.isalpha():
        continue
    # písmena zahrň do slovníku
    if znak not in statistika:
        statistika[znak] = [1, 0]
    else:
        statistika[znak][0] += 1

pismen = len(text)
for klic in statistika:
    statistika[klic][1] = statistika[klic][0] / pismen
    # převod na procenta
    statistika[klic][1] = 100 * statistika[klic][1]

pprint.pprint(statistika)
```

3. Upravte předchozí program tak, aby nerozlišoval mezi malými a velkými písmeny.

[-] řešení (*typy/slovniky/count_letters.3.py*)

```
import pprint

text = ''
with open('leacock-abc.txt', mode='r', encoding='utf-8') as
f:
    text = f.read().lower()

statistika = {}
for znak in text:
    # přeskoč nepísmenné znaky
    if not znak.isalpha():
        continue
    # písmena zahrň do slovníku
    if znak not in statistika:
        statistika[znak] = [1, 0]
    else:
        statistika[znak][0] += 1

pismen = len(text)
for klic in statistika:
    statistika[klic][1] = statistika[klic][0] / pismen
    # převod na procenta + zaokrouhlení
    statistika[klic][1] = 100 * statistika[klic][1]

pprint.pprint(statistika)
```

4. Obměňte první program tak, aby ukládal počet výskytů nikoli písmen, ale slov.

[-] řešení (*typy/slovniky/count_words.1.py*)

```
import string
import pprint

text = ''
with open('leacock-abc.txt', mode='r', encoding='utf-8') as
f:
    text = f.read().split()
text = [ slovo.strip(string.punctuation) for slovo in text ]

statistika = {}
for slovo in text:
    if slovo not in statistika:
        statistika[slovo] = 1
    else:
        statistika[slovo] += 1

pprint.pprint(statistika)

# pro zajímavost ještě seřazeno podle počtu výskytů
print( sorted(statistika, key=lambda x: statistika[x],
reverse=True) )
```

5. Rozšiřte předchozí program tak, že hodnotou bude n-tice o dvou prvcích – prvním bude počet výskytů daného slova v textu, druhým jeho relativní četnost v % (vzhledem k počtu všech slov v textu).

[-] řešení (*typy/slovniky/count_words.2.py*)

```
import string
import pprint

text = ''
with open('leacock-abc.txt', mode='r', encoding='utf-8') as
f:
    text = f.read().split()
text = [ slovo.strip(string.punctuation) for slovo in text ]

statistika = {}
for slovo in text:
    if slovo not in statistika:
        statistika[slovo] = [1, 0]
    else:
        statistika[slovo][0] += 1

slov = len(text)
for klic in statistika:
    statistika[klic][1] = statistika[klic][0] / slov
    # převod na procenta
    statistika[klic][1] = 100 * statistika[klic][1]

pprint.pprint(statistika)
```

6. Napište funkci, která spojí dohromady dva slovníky.
Ukázka: `{"a": 1, "b": 2, "c": 3}, {"d": 7, "e": 8} => {"a": 1, "b": 2, "c": 3, "d": 7, "e": 8}`

[-] řešení (*typy/slovniky/01a.py*)

```
def concat_dicts(d1, d2):
  d = {}
  for k,v in d1.items():
    d[k] = v
  for k,v in d2.items():
    d[k] = v
  return d

# test
print( concat_dicts( {"a":1, "b":2, "c":3}, {"d":7, "e":8} )
)
```

[-] řešení (*typy/slovniky/01b.py*)

```
def concat_dicts(d1, d2):
  d = {}
  d.update(d1)
  d.update(d2)
  return d

# test
print( concat_dicts( {"a":1, "b":2, "c":3}, {"d":7, "e":8} )
)
```

7. Napište funkci, která bude „obracet“ slovník. Bude brát slovník jako jediný argument a vrátí nový slovník, ve kterém budou hodnoty ze vstupního slovníku převedeny na klíče a klíče na hodnoty.
Ukázka: `{1: 'A', 2: 'B'} => {'A': 1, 'B': 2}`

[-] řešení (*typy/slovniky/02.py*)

```
def invert_dict(dictionary):
  ret = {}
  for k, v in dictionary.items():
    ret[v] = k
  return ret

# test
d1 = { 1:'A', 2:'B', }
d2 = { 1:'A', 2:'B', 3:'B', 4:'A', 5:'C', }

print(d1)
print( invert_dict(d1) )

print()

print(d2)
print( invert_dict(d2) )
```

8. Upravte předchozí program tak, aby bral v potaz skutečnost, že hodnoty nemusí být unikátní. Pro každou hodnotu tedy bude vytvářet seznam původních klíčů.
Ukázka: `{1: 'A', 2: 'B', 3: 'B', 4: 'A', 5: 'C'} => {'A': [1,4], 'B': [2,3], 'C': [5]}`

[-] řešení (*typy/slovniky/03.py*)

```
def invert_dict_multi(dictionary):
  """handles multiple different keys for one value"""

  ret = {}
  for k, v in dictionary.items():
      if v in ret:
          ret[v].append(k)
      else:
          ret[v] = [k]
  return ret


# test
d1 = { 1:'A', 2:'B', }
d2 = { 1:'A', 2:'B', 3:'B', 4:'A', 5:'C', }

print(d1)
print( invert_dict_multi(d1) )

print()

print(d2)
print( invert_dict_multi(d2) )
```