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

Generátorová notace

1. Napište generovaný seznam pro následující operace se seznamem *xs = [1, 4, 2, 67, 2, -90, 456, 2, 1, 45, 5, 6, 7, 9, -3, 2, 4, 5, 61]*:

- převeďte každé číslo na jeho třetí mocninu (x^3)
- převeďte každé kladné číslo na jeho dekadický logaritmus (log x)
- vyberte pouze x dělitelná 2 nebo 3
- převeďte každé číslo na řetězec (číslo by mělo být zarovnané doprava a zabírat minimálně 4 znaky)
- vyberte pouze taková čísla, jejichž druhá mocnina je větší než 100
- pro každé x menší než 100 vytvořete n-tici (tuple) (x, x^2, x^3)

[-] řešení (*comprehension/01.py*)

```
xs = [1, 4, 2, 67, 2, -90, 456, 2, 1, 45, 5, 6, 7, 9, -3, 2,
4, 5, 61,]
print(xs)
print()

# convert every number to its third power (x<sup>3</sup>)
print( [x**3 for x in xs] )

# convert every number to its decadic logarithm (log x),
# but select only xs that are positive (>0)
import math
print( [math.log10(x) for x in xs if x > 0] )

# select only x that are divisible by 2 or 3
print( [x for x in xs if x % 2 == 0 or x % 3 == 0] )

# convert every number to string. The number should be
right-aligned and occupy at
# least 4 characters (spaces should be used to fill the
empty space :)
print()
#print( ["%4d" % x          for x in xs] )
print( ["{:>4}".format(x) for x in xs] )

# select only x whose second power is larger that 100
print( [x for x in xs if x**2 > 100] )

# create a tuple of (x, x<sup>2</sup>, x<sup>3</sup>)
# for each x that is smaller than 100
print( [(x, x**2, x**3) for x in xs if x < 100] )
```

2. Napište generovaný seznam pro následující operace se seznamem *ws =*

["dog", "pig", "hippo", "dogs", "tyranosaurus", "human", "shark", "lion"]:

- převeďte každé slovo na velká písmena
- vyberte a kapitalizujte každé slovo, které má víc než 3 písmena
- vyberte všechna slova a kapitalizujte každé slovo, které má víc než 3 písmena
- vyberte pouze slova, která končí na „s“
- převeďte každé slovo na jeho délku (např. "lion"→4)

[-] nápověda

```
x.capitalize() if len(w)>3 else w
```

[-] řešení (*comprehension/02.py*)

```
ws=["dog", "pig", "hippo", "dogs", "tyranosaurus", "human",
"shark", "lion",]
print(ws)
print()

# convert every word to upper-case
print( [w.upper() for w in ws] )

# capitalize every word, but select only those that are
longer that 3 letters
print( [w.capitalize() for w in ws if len( w) > 3] )

# capitalize only those words, that are longer that 3
letters
print( [w.capitalize() if len( w) > 3 else w for w in ws] )

# select only the words that end with an "s"
print( [w for w in ws if w[-1] == 's'] )
print( [w for w in ws if w.endswith('s')] )

# convert each word to its length ("lion"->4)
print( [len(w) for w in ws] )
```

3. Vytvořte pomocí generátorové notace slovník slov, vyskytujících se v zadaném textu, přičemž slova budou klíčem a hodnotou bude:

- počet znaků ve slově
- počet výskytů slova v textu
- n-tice obou předchozích údajů

[-] řešení (*comprehension/03.py*)

```
with open('example.2.txt', 'r', encoding='utf-8') as f:
    text = f.read()

#print(text)
slova = text.split()

# počet znaků ve slově
print( { slovo: len(slovo) for slovo in slova } )

# počet výskytů slova v textu
print( { slovo: slova.count(slovo) for slovo in slova } )

# n-tice obou předchozích údajů
print( { slovo: (len(slovo), slova.count(slovo)) for slovo
in slova } )
```

4. Generátorovou notací pro množinu zjistěte, z jakých *znaků* se skládá zadaný kratší text.

[-] řešení (*comprehension/04.py*)

```
text = """Řádka se čtyřmi slovy.
Tahle řádka má slov pět.

Další řádka, tentokrát se šesti slovy.
A jiná, kde těch slov je sedm.

A potřebujeme znovu pět slov.
Co si takhle zopakovat slov šest."""

znaky = { ch for ch in text }
print(znaky)
```

5. Generátorovou notací pro množinu zjistěte, z jakých *slov* se skládá zadaný delší text.

[-] řešení (*comprehension/05.py*)

```
text = """Řádka se čtyřmi slovy.
Tahle řádka má slov pět.

Další řádka, tentokrát se šesti slovy.
A jiná, kde těch slov je sedm.

A potřebujeme znovu pět slov.
Co si takhle zopakovat slov šest."""

# s interpunkcí
slova_s_interpunkci = { slovo for slovo in text.split() }
print(slova_s_interpunkci, len(slova_s_interpunkci))

# bez interpunkce
import string
slova = { slovo.strip(string.punctuation) for slovo in
slova_s_interpunkci }
print(slova, len(slova))
```

6. Generátorovou notací pro množinu zjistěte, jak dlouhá slova se v textu vyskytují.

[-] řešení (*comprehension/06.py*)

```
import string

text = """Řádka se čtyřmi slovy.
Tahle řádka má slov pět.

Další řádka, tentokrát se šesti slovy.
A jiná, kde těch slov je sedm.

A potřebujeme znovu pět slov.
Co si takhle zopakovat slov šest."""

slova_s_interpunkci = { slovo for slovo in text.split() }
slova = { slovo.strip(string.punctuation) for slovo in
slova_s_interpunkci }
delka_slov = { len(slovo) for slovo in slova }

print(delka_slov)
```

Autorem příkladů 1 a 2 a jejich původního řešení pro Python 2.x je Bedřich Košata.