Knihovny Python

1. Opakování základů jazyka (datové struktury, syntaxe) (06.03.2023)

Přirozené datové typy

1. Boolean
2. Čísla (celá, desetinná, zlomky, komplexní)
3. Řetězce
4. Bajty a pole bajtů
5. Seznamy
6. N-tice
7. Množiny
8. Slovníky

Boolean

-

In [1]:
```
bool()
```

Out[1]:
```
False
```

- datový typ reprezentují pouze dvě hodnoty (False a True)
- výsledkem podmínky je právě typ bool

In [2]:
```
True and False or False
```

Out[2]:
```
False
```

In [3]:
```
1 > 0
```

Out[3]:
```
True
```

In [4]:
```
'a' in 'ahoj'
```

Out[4]:
```
True
```

In [5]:
```
'polo' not in 'poloostrov'
```

Out[5]:
```
False
```

- booleanovské operátory

In [6]:
```
1 == 0 or 3 > 2 and 4 < 1 or 3 <= 5 and 0.0 != 0.2
```

Out[6]: True

Čísla

-

- pro naše potřeby si vystačíme s celými (int) a desetinnými čísly (float)

In [7]:
```
int()
```

Out[7]: 0

In [8]:
```
float()
```

Out[8]: 0.0

- s čísly lze dělat běžné matematické operace

In [9]:
```
1 + 2 # sčítání
```

Out[9]: 3

In [10]:
```
3 - 0.5  # odčítání
```

Out[10]: 2.5

In [11]:
```
0.2 * 5  # násobení
```

Out[11]: 1.0

In [12]:
```
9 / 3  # dělení
```

Out[12]: 3.0

In [13]:
```
9 // 4 # celočíselné dělení
```

Out[13]: 2

In [14]:
```
2 ** 2 # umocnění
```

In [15]:
```
8 % 3  # modulo
```

- chování čísel v booleanovských výrazech

In [16]:
```
bool(0.0)
```

In [17]:
```
bool(-3)
```

In [18]:
```
0.5 == 1 / 2
```

Řetězce

-

In [19]:
```
str()
```

- posloupnost znaků v Unicode

- skládání řetězců

In [29]:
```
pozdrav = 'ahoj'
```

In [30]:
```
jmeno = 'Davide'
```

In [31]:
```
pozdrav + ' ' + jmeno
```

Out[31]:
```
'ahoj Davide'
```

- z řetězce je možné dělat řezy, vybírat jednotlivé znaky, procházet je

In [32]:
```
retezec = 'neuvěřitelné, všechno se dá popsat pomocí trojúhelníků, myšleno ironi
```

In [33]:
```
retezec[14:54]
```

Out[33]:
```
'všechno se dá popsat pomocí trojúhelníků'
```

In [34]:
```
retezec[53]
```

Out[34]:
```
'ů'
```

In [35]:
```
from string import ascii_lowercase
for znak in ascii_lowercase:
    print(znak)
```

```
a
b
c
d
e
f
g
h
i
j
k
l
m
n
o
p
q
r
s
t
u
v
w
x
y
z
```

- na pozice řetězce nelze zapisovat

In [36]:
```
nazev = 'Dobrou noc'
```

In [37]:
```
nazev[10] = 'n'
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[37], line 1
----> 1 nazev[10] = 'n'

TypeError: 'str' object does not support item assignment
```

- nahrazení v řetězci lze provést jinými způsoby

In [38]:
```
nazev[:7] + 'm' + nazev[8:]
```

Out[38]:
```
'Dobrou moc'
```

In [39]:
```
nazev.replace('Dobrou', 'Nesmírnou')
```

Out[39]:
```
'Nesmírnou noc'
```

- pro vyhledávání, případně nahrazování lze využít regulární výrazy (pouze ukázka v běhu)

In [40]:
```
slozitost = 'hledáš snad nějakou emailovou@adresu.cz?'
```

In [41]:
```
import re
re.search(r'[a-zA-Z0-9._-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,4}', slozitost).group(0)
```

Out[41]:
```
'emailovou@adresu.cz'
```

- pomocí metody *len* je možné zjistit počet znaků v řetězci

In [42]:
```
slovo = 'nejkulaťoulinkatější'
```

In [43]:
```
len(slovo)
```

Out[43]:
```
20
```

- řetězce lze jednoduše modifikovat

In [44]:
```
prohlaseni = 'nemám rád šmouly!'
```

In [45]:
```
prohlaseni.capitalize()
```

Out[45]:
```
'Nemám rád šmouly!'
```

In [46]:
```
prohlaseni.upper()
```

Out[46]:
```
'NEMÁM RÁD ŠMOULY!'
```

In [47]:
```
prohlaseni.lower()
```

Out[47]:
```
'nemám rád šmouly!'
```

In [48]:
```
az_prilis_mnoho_mezer = '                                        hle!
```

In [49]:
```
az_prilis_mnoho_mezer.strip()
```

Out[49]:
```
'hle!'
```

- řetězce lze rozdělovat, výsledkem je seznam

In [50]:
```
mnoho_slov = 'jestli snad jsem někdy řekl mnoho slov'
```

In [51]:
```
mnoho_slov.split('j')
```

Out[51]:
```
['', 'estli snad ', 'sem někdy řekl mnoho slov']
```

- formátování rětězce pomocí format a F-string

In [52]:
```
jmeno = 'Eliška'
```

In [53]:
```
f'Tvoje jméno je {jmeno}!'
```

Out[53]:
```
'Tvoje jméno je Eliška!'
```

In [54]:
```
veta = 'Tvoje jméno je {jmeno}!'
```

In [55]:
```
veta.format(jmeno=jmeno)
```

Out[55]:
```
'Tvoje jméno je Eliška!'
```

Bajty

-

- například soubor s obrázkem ve formátu jpeg
- kódování řetězce je převod z posloupnosti znaků do posloupti bajtů

In [56]:
```
jmeno = 'Alžběta'
```

In [57]:
```
jmeno.encode('utf-8')
```

Out[57]:
```
b'Al\xc5\xbeb\xc4\x9bta'
```

Seznamy

-

In [58]:
```
list()
```

Out[58]:
```
[]
```

- uložení více prvků (hodnot či proměnných) do jediné proměnné

- lze využít zápisu pomocí klíčového slova list nebo pomocí literáru []

In [59]:
```
seznam_a = list()
```

In [60]:
```
seznam_b = []
```

- přídávání prvku do seznamu pomocí metody *append*

In [61]:
```
seznam_a.append('a')
```

In [62]:
```
seznam_a
```

Out[62]:
```
['a']
```

- přídávání prvku do seznamu pomocí metody *insert*

In [63]:
```
seznam_a.insert(1, 'y')
```

In [64]:
```
seznam_a
```

```
Out[64]: ['a', 'y']
```

- přídávání seznamu k seznamu pomocí metody *extend*

```
In [65]: seznam_a.extend(['b', 'c', 'd'])
```

```
In [66]: seznam_a
```

```
Out[66]: ['a', 'y', 'b', 'c', 'd']
```

- stejně jako je možné v řetězci zjistit počet znaků, je možné v seznamu zjistit počet prvků pomocí metody *len*

```
In [67]: len([1, None, [0, 1], 2, 4, 'š', 'konec'])
```

```
Out[67]: 7
```

- jistě jste si všimli, že seznam nemusí být honogenní, dokonce může obsahovat jiné seznamy (a další struktury)

- řezání seznamu

```
In [68]: seznam = [1, 2, 3, 4, 5]
```

```
In [69]: seznam[2:4]  # přímo
```

```
Out[69]: [3, 4]
```

```
In [70]: řez = slice(2, 4)
```

```
In [71]: seznam[řez]  # pomocí řezu
```

```
Out[71]: [3, 4]
```

- vyhledávání hodnoty v seznamu

```
In [72]: seznam = ['jedna', 'dva', 'tři', 'čtyři', 'pět', 'dva']
```

```
In [73]: seznam.count('dva')  # pomocí metody count zjišťujeme počet uvedeného prvku v se
```

```
Out[73]: 2
```

```
In [74]: seznam.index('dva')  # vyhledávání pomocí metody index vrací první nalezený výsk
```

```
Out[74]: 1
```

```
In [75]: seznam.index('šest')  # co se ale stane, když se prvek v seznamu nenajde?
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[75], line 1
----> 1 seznam.index('šest')  # co se ale stane, když se prvek v seznamu nenajd
e?

ValueError: 'šest' is not in list
```

In [76]:
```
'jedna' in seznam  # pomocí in
```

Out[76]:
```
True
```

- odstraňování položek ze seznamu

In [77]:
```
del seznam[1]  # v uvedením indexu
```

In [78]:
```
seznam
```

Out[78]:
```
['jedna', 'tři', 'čtyři', 'pět', 'dva']
```

In [79]:
```
seznam.remove('tři')  # s vyhledáním hodnoty
```

In [80]:
```
seznam
```

Out[80]:
```
['jedna', 'čtyři', 'pět', 'dva']
```

In [81]:
```
prvek = seznam.pop(0) # pomocí metody pop
```

In [82]:
```
prvek
```

Out[82]:
```
'jedna'
```

- vlastnosti seznamu v booleanovských výrazech

In [83]:
```
bool([])  # výsledek?
```

Out[83]:
```
False
```

N-tice

-

- v jistém ohledu jsou velmi podobné seznamům

- nicméně jsou neměnitelné

- definice pomocí metody tuple nebo ()

In [84]:
```
tuple([2, 3, 4])
```

Out[84]: (2, 3, 4)

In [85]:
```
(2, 3, 4)
```

Out[85]: (2, 3, 4)

úkol za zlatého bludišťáka; vytvořte jednoprvkovou n-tici

In [86]:
```
jednice = (1, )
```

In [87]:
```
len(jednice)
```

Out[87]: 1

Množiny

-

- neuspořádané kolekce jedinečných hodnot

- definované pomocí hodnot v {} nebo pomoc9 metody set()

In [88]:
```
{1, 2, 3}
```

Out[88]: {1, 2, 3}

In [89]:
```
set((1, 2, 3, 3, 5, 4, 1, 2, 3))
```

Out[89]: {1, 2, 3, 4, 5}

In [90]:
```
type({})  # copak vytvoříme z prázdných složených závorek?
```

Out[90]: dict

- přidávání prvků do množiny

In [91]:
```
mnozina = {1, 2, 3, 4, 5, 6}
```

In [92]:
```
mnozina.add(0)  # přidání prvku
```

In [93]:
```
mnozina
```

Out[93]: {0, 1, 2, 3, 4, 5, 6}

In [94]:
```
mnozina.update((8, 9, 1, 2, 3))
```

In [95]:
```
1 in mnozina
```

Out[95]: True

- odebírání prvků z množiny

In [96]:
```
mnozina.discard(1)
```

In [97]:
```
mnozina.add(1)
```

In [98]:
```
mnozina.remove(2)
```

In [99]:
```
mnozina.pop()  # stejně jako u seznamu, s jednou výjimkou
```

Out[99]:
```
0
```

- množiny jsou neuspořádané, proto je vrácená hodnora náhodná

In [100…
```
mnozina.clear()
```

In [101…
```
mnozina
```

Out[101]:
```
set()
```

- běžné množinové operace

In [102…
```
mnozina_a = {1, 3, 5, 7, 9}
mnozina_b = {2, 4, 6, 8}
```

In [103…
```
mnozina_a.union(mnozina_b)  # sjednocení
```

Out[103]:
```
{1, 2, 3, 4, 5, 6, 7, 8, 9}
```

In [104…
```
mnozina_a.difference(mnozina_b) # je v A, ale není v B
```

Out[104]:
```
{1, 3, 5, 7, 9}
```

In [105…
```
mnozina_a.intersection(mnozina_b) # průnik
```

Out[105]:
```
set()
```

In [106…
```
mnozina_a.symmetric_difference(mnozina_b) # je v A, ale není v B, nebo je v B, a
```

Out[106]:
```
{1, 2, 3, 4, 5, 6, 7, 8, 9}
```

Slovníky

-

- neuspořádaná kolekce dvojic klíč-hodnota

- definice pomocí metody dict() nebo složených závorek

In [107…
```
dict()
```

Out[107]:
```
{}
```

In [108…
```
{'a': 1, 'b': 2}
```

Out[108]:
```
{'a': 1, 'b': 2}
```

- úprava slovníku

In [109…
```
hodnoceni = {'Karel': 2, 'Jana': 1, 'Lenka': 2, 'Jiří': None}
```

In [110…
```
hodnoceni['Jan'] = 3
```

In [111…
```
hodnoceni
```

Out[111]:
```
{'Karel': 2, 'Jana': 1, 'Lenka': 2, 'Jiří': None, 'Jan': 3}
```

- slovníky lze také vnořovat, nebo jako prvky využít struktury

In [112…
```
hodnoceni = {'Filip': [1, 1, 2, 4, 2, 1], 'Anna': [3, 1, 1, 2, 2, 1], 'Ludmila':
```

In [121…
```
# na řadek vypsat jméno a za dvojtečkou vypsat známky oddělené čárkou, chceme vy
for jmeno, znamky in hodnoceni.items():
    print(jmeno, ', '.join([str(znamka) for znamka in znamky if znamka]))
```

```
Filip 1, 1, 2, 4, 2, 1
Anna 3, 1, 1, 2, 2, 1
Ludmila 3, 1, 2, 2, 1
```

In [114…
```
seznam = ['ahoj', 'jak', 'se', 'mas', '?']
```

In [115…
```
list(map(len, seznam))
```

Out[115]:
```
[4, 3, 2, 3, 1]
```

In [116…
```
list(filter(lambda x: len(x) < 4, seznam))
```

Out[116]:
```
['jak', 'se', 'mas', '?']
```

In [117…
```
for no, znak in enumerate('ahoj', start=1):
    print(no, znak)
```

```
1 a
2 h
3 o
4 j
```

- comprehension

In [123…
```
[str(i) for i in range(100) if i % 3]  # seznamová
```

Out[123]:
```
{1: 'I', 2: 'II', 3: 'III', 4: 'IV', 5: 'V'}
```

In [ ]:
```
{roman_number: latin_number for roman_number, latin_number in zip([1, 2, 3, 4, 5
```

In [128…
```
for percent in (i/100 for i in range(0, 101, 20)):
```

```
    print(percent)
```

```
0.0
0.2
0.4
0.6
0.8
1.0
```