Řetězce

December 6, 2023

1 Řetězce (string)

Základy práce s řetězci (slova, věty, texty).

Velmi podobné manipulaci se seznamy. Hlavní rozdíl: řetězec se nedá přepisovat znak po znaku (tzv. immutable object).

[1]:
```
slovo = 'python'
slovo2 = "Python"
```

[2]:
```
len(slovo)       # pocet znaku
```

[2]:
```
6
```

[3]:
```
slovo[2]         # znak na 3. miste
```

[3]:
```
't'
```

[4]:
```
slovo[:3] + 'el'         # rezy funguji, retezce muzeme scitat
```

[4]:
```
'pytel'
```

[5]:
```
slovo = 'nove slovo'     # muzeme "prepsat" cely retezec
print(slovo)
```

```
nove slovo
```

[6]:
```
slovo[1] = 't'           # nemuzeme menit retezec po znacich
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[6], line 1
----> 1 slovo[1] = 't'

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

[7]:
```
for p in slovo:          # prochazeni retezce po jednotlivych znacich
    print(p)
```

```
n
o
v
e

s
l
o
v
o
```

[18]:
```
seznam = list(slovo)
seznam[4] = '+'
print(seznam)
```

```
['n', 'o', 'v', 'e', '+', 's', 'l', 'o', 'v', 'o']
```

[22]:
```
retezec = "".join(seznam)
print(retezec)
```

```
nove+slovo
```

[20]:
```
print("-".join(seznam))
```

```
n-o-v-e-+-s-l-o-v-o
```

1.1 Operace specifické pro řetězce

Malá ukázka, více najdete v dokumentaci (help).

[8]:
```
'kapitalky'.upper()
```

[8]:
```
'KAPITALKY'
```

[9]:
```
'MALE'.lower()
```

[9]:
```
'male'
```

[27]:
```
odpoved = input('ano/ne? ')
if odpoved.lower() == 'ano':
    print('Ano')
else:
    print('Ne')
```

```
ano/ne? Ne
Ne
```

[28]:
```
'schovavana'.find('va')       # Vraci index prvniho vyskytu
```

[28]:
```
4
```

[11]:
```
'schovavana'.find('ananas')   # -1 pokud podretezec nenajde
```

[11]:
```
-1
```

[12]:
```
'Rozdel tuto vetu.'.split()   # rozdeleni po slovech; split: casto pouzivane ...
```

[12]:
```
['Rozdel', 'tuto', 'vetu.']
```

[30]:
```
rada = input('Zadejte cisla na jednu radku: ')
cisla = rada.split()
cisla2 = [int(p) for p in cisla]     # Priste (list comprehensions)
print(cisla2)
```

```
Zadejte cisla na jednu radku: 1 4 6  8
[1, 4, 6, 8]
```

[31]:
```
'1+2+3'.split(sep='+')       # rozdeleni se separatorem +
```

[31]:
```
['1 ', '2', '3']
```

[14]:
```
'+'.join(['1','2','3'])      # spojeni znakem +
```

[14]:
```
'1+2+3'
```

[ ]:
```
help(str)
```

[ ]:
```
help('')
```

Úkol Hra šibenice – umrlce najdete níže

Další úkoly Napište funkci, která
- spočítá, kolik zadaný řetězec obsahuje slov (oddělených mezerami)
- spočítá, kolik různých slov se vyskytuje v zadaném řetězci. Ošetřete tak, aby např. 'Test' a 'test' bylo považováno za stejné slovo.
- vyhodnotí výraz se sčítáním. Příklad: vstup z klávesnice bude 12+34+1 (jeden výraz), výstup bude číslo 47. Použijte funkci split se separátorem.

[ ]:
```
sibenice = [
"              \n              \n              \n              \n              ␣
 ↪\n______________",
"              \n              \n              \n              \n              ␣
 ↪\n___|__________",
"              \n              \n              \n              \n   |          ␣
 ↪\n___|__________",
"              \n              \n              \n   |          \n   |          ␣
 ↪\n___|__________",
```

```
"             \n             \n   |         \n   |         \n   |         ␣
 ↪\n___|_________",
"             \n   |         \n   |         \n   |         \n   |         ␣
 ↪\n___|_________",
"   _         \n   |         \n   |         \n   |         \n   |         ␣
 ↪\n___|_________",
"   __        \n   |         \n   |         \n   |         \n   |         ␣
 ↪\n___|_________",
"   ___       \n   |         \n   |         \n   |         \n   |         ␣
 ↪\n___|_________",
"   ____      \n   |         \n   |         \n   |         \n   |         ␣
 ↪\n___|_________",
"   _____     \n   |         \n   |         \n   |         \n   |         ␣
 ↪\n___|_________",
"   _____     \n   |    |    \n   |         \n   |         \n   |         ␣
 ↪\n___|_________",
"   _____     \n   |    |    \n   |    O    \n   |         \n   |         ␣
 ↪\n___|_________",
"   _____     \n   |    |    \n   |    O    \n   |   -     \n   |         ␣
 ↪\n___|_________",
"   _____     \n   |    |    \n   |    O    \n   |   -|    \n   |         ␣
 ↪\n___|_________",
"   _____     \n   |    |    \n   |    O    \n   |   -|-   \n   |         ␣
 ↪\n___|_________",
"   _____     \n   |    |    \n   |    O    \n   |   -|-   \n   |   /     ␣
 ↪\n___|_________",
"   _____     \n   |    |    \n   |    O    \n   |   -|-   \n   |   / \ ␣
 ↪\n___|_________"
]
```

[ ]: