Řetězec a operace s řetězci

Řetězec

Řetězec (*string*) je reprezentace textových informací. Python nemá na rozdíl od jiných jazyků typ znak (*char*), pouze řetězec délky jedna.

Řetězce jsou uzavřeny do apostrofů nebo uvozovek, víceřádkové řetězce se uzavírají do trojic uvozovek či trojic apostrofů.

Řetězce patří mezi konstantní posloupnosti (*immutable sequences*), což znamená, že s nimi můžeme pracovat jako s ostatními posloupnostmi, avšak nemůžeme měnit jednotlivé prvky (znaky). Operace s řetězci jsou obecnější a platí pro všechny posloupnosti.

Prázdné řetězce zapisujeme pomocí prázdného páru apostrofů či uvozovek. Příklad:

```
>>> a=""
>>> a
''
```

Python poskytuje **dva typy řetězcových objektů:**

- **standardní řetězce** jsou posloupnosti bajtů obsahující **osmibitová** data. Příklad:

```
>>> "ahoj"
'ahoj'
```

- **řetězce Unicode,** což jsou posloupnosti **šestnáctibitových** znaků. Řetězce Unicode jsou uvozeny znakem **u.** Příklad:

```
>>> u"ahoj"
u'ahoj'
```

Operace s řetězci

Délka řetězce

Délku řetězce vrací **funkce len**, jejímž argumentem je řetězec. Příklad:

```
>>> len("pitomec")
7
```

Skládání řetězců

Skládat řetězce je možno pomocí operátoru +. Příklad:

```
>>> a = "jazyk"
>>> b = "Python"
>>> a + " " + b
'jazyk Python'
```

Opakování řetězců

Opakování řetězců zajišťuje operátor * následovaný číslem, které uvádí počet opakování. Příklad:

```
>>> "Python " * 4
'Python Python Python Python '
```

Operátor in

Pomocí operátoru in lze procházet jednotlivými znaky řetězce či testovat přítomnost znaku v řetězci. Příklady:

```
>>> a = "pitomec"
>>> for b in a: print b,                                    #procházíme
posloupností
p i t o m e c
>>> "t" in a                                  # 1 znamená pravda
1
```

Výběr části řetězce

K jednotlivým částem řetězce lze přistupovat pomocí operátoru [i], psaného za řetězec, kde i je pořadí znaku v řetězci. Číslování znaků v řetězci začíná 0 a končí na „délka řetězce mínus jedna“. Znaky lze číslovat i zápornými čísly. Záporná čísla zde mají význam „délka plus záporné číslo“ nebo jako vzdálenost od konce (resp. zprava). Poslední znak řetězce má číslo -1.

Např. Mějme řetězec ‘spam’, který je délky čtyři. Znak s indexem 0 je „s“ první znak zleva), znak s indexem jedna je „p“ druhý znak zleva), poslední znak řetězce „m“ má index 3, tj. délka řetězce mínus 1. Předposlední znak řetězce „a“ má index 2, ale také -2, protože „délka plus záporné číslo“ je [4+(-2)] tedy 2.

Python umožňuje vybírat část řetězce dvojicí pořadí [i : j], přičemž výběr bude začínat na pozici i (pořadím uvedeným zleva) a pokračuje všemi dalšími znaky až po znak j (pořadím uvedeným vpravo), přičemž znak na pozici j již do výběru zahrnut nebude.

Shrnutí:

Výběr prvku – S[i]

- vrací kopii prvku na daném pořadí (číslováno od nuly)
- záporné hodnoty se přičítají k délce posloupnosti (a jejich absolutní hodnota je vzdálenost od konce, zprava). Prvek první zprava má index [-1].
- S[0] vrací hodnotu prvního prvku.
- S[-2] vrací hodnotu předposledního prvku (tj. ekvivalent S[len(S) -2]

Výběr části- S[i : j]

- Vrací kopii souvislé části (popř. celé původní) posloupnosti
- Implicitní hodnotou (tj, pokud žádnou neuvedeme) i je 0 a j len(S)
- S[1 : 3] vrací kopii druhého a třetího prvku.
- S[1 :] vrací kopii od druhého znaku do konce (len(S))
- S[: -1] vrací kopii prvků od začátku po poslední znak (na pořadí len(S) -1), který však již ve výsledku obsažen nebude.

Příklady:

```
>>> a = "pitomec"
>>> a[ : ]
'pitomec'
>>> a[1 : 3]
'it'
>>> a[4]
'm'
>>> a[ : -2]
'pitom'
>>> a[2 : ]
'tomec'
>>> a[-4 : ]
'omec'
```

Změny a formátování řetězců

Řetězce nelze měnit, takový pokus vyvolá výjimku, jak ukazuje následující příklad:

```
>>> S = "spam"
>>> S[0] = "x"
Traceback (most recent call last):
```

```
    File "<pyshell#1>", line 1, in ?
      S[0] = "x"
  TypeError: object doesn't support item assignment
  Pomocí formátování řetězců lze vytvářet nové změněné řetězce.
Příklad:
  >>> S = "ham"
  >>> S = S + "burger"
  >>> S
  'hamburger'
```

Základem je operátor %s, který převádí libovolnou hodnotu na řetězec. V tabulce jsou další (méně užívané operátory):

%c řetězec délky jedna (*char*, znak)	%f reálné číslo (dále RČ), plný tvar
%i celé číslo (*integer,* dále CČ, nebo %d)	%e RČ, exponenciální tvar (s malým e)
%u CČ bez znaménka (*unsigned*)	%E RČ, exponenciální tvar (s velkým E)
%o CČ, osmičková (*octal*) reprezentace	%g RČ, automatická volba (s malým e)
%x CČ, šestnáctková *(hex*) reprezentace (a-f)	%G RČ, automatická volba (s velkým E)
%X CČ, šestnáctková (*hex*) reprezentace (A-F)	%% samotný znak procento

Příklady:

```
>>> import math
>>> a = 0.1; x = 4*math.pi * 1e-7          #permeabilita vakua
>>> a
0.10000000000000001
>>> x
1.2566370614359173e-006
>>> "%e" % (x)
'1.256637e-006'
>>> "%g" % (x)
'1.25664e-006'
>>> "%G" % (x)
'1.25664E-006'
>>> "%.4f" % (a*10)
'1.0000'
>>> "%.4d" % (a*10)
'0001'
>>> "%g" % (a)
'0.1'
```

Literatura:

[1] Rubeš, J.: Nebojte se programovat, ComputerMedia, Bedihošť 2001
[2] Lutz, M., Ascher, D.: Naučte se Python, Grada, Praha 2003
[3] Beazley, D. M.: Python, Neocortex, Praha 2002
[4] Python Reference Manual
[5] Švec, J.: Létající circus, Python tutoriál, 2003