PYTHON

Textové (znakové) reťazce

Vypracovala: Ing. Eva Gabonayová
Predmet: Informatika
Vzdelávacia oblasť: Matematika a práca s informáciami
Dátum spracovania: 15. 11. 2020

Typ string

- reťazec je postupnosť znakov uzavretá v apostrofoch **' '** alebo v úvodzovkách **" "**
- vieme priradiť reťazec do premennej
- zreťaziť (spojiť) dva reťazce **+**
- násobiť (zlepiť viac kópií) reťazca *****
- načítať zo vstupu (pomocou **input()**) a vypisovať (pomocou **print()**)
- vyrobiť z čísla reťazec (**str()**), z reťazca číslo (**int()**, **float()**)

Typ premennej

- Použijeme funkciu **type (meno_premennej)**

```
>>> abc = 'Monty Python'
>>> type(abc)
<class 'str'>                reťazec
>>> a=12
>>> type(a)
<class 'int'>                celé číslo
>>> b=3.8
>>> type(b)
<class 'float'>              desatinné číslo
>>> input(c)
```

Zreťazenie reťazcov operáciou +

```
slovo = 'Py'+'thon'
print(slovo)
```

```
Python
>>>
```

Viacnásobné zreťazenie reťazcov operáciou *

```
pocet = '12'
vysledok = pocet*5
print(vysledok)
```

```
1212121212
>>>
```

Funkcia int(*reťazec*)

- z reťazca vytvorí číslo (ak obsahuje len znaky, ktoré môže obsahovať číslo)

```
a = '1250'
b = int(a)
b = b+10
print(b)
```

```
1260
>>>
```

Funkcia str(*číslo*)

- z čísla vytvorí textový reťazec

```
body = 12
oznam = 'Tvoj počet bodov:'+str(body)
print(oznam)
```

```
Tvoj počet bodov:12
>>>
```

Funkcia input(*reťazec*)

- načítanie vstupu (program čaká na zadanie vstupu od používateľa)

```
meno = input('Ako sa voláš?')
print('Ahoj '+ meno +'  :)')
```

```
Ako sa voláš?Eva
Ahoj Eva  :)
>>>
```

Použitie for cyklu

- na postupné prechádzanie reťazcom po jednotlivých znakoch

```
for znak in 'Python':
    print(znak)
```

```
P
y
t
h
o
n
>>>
```

```
retazec = 'Python'
for znak in retazec:
    print(znak)
```

Funkcia len(*reťazec*)

- zistí dĺžku reťazca

```
slovo = input('Napíš slovo:')
dlzka = len(slovo)
print('Počet znakov v slove je:', dlzka)
```

```
Napíš slovo:Lokomotíva
Počet znakov v slove je: 10
>>>
```

```
Napíš slová:Poprad je krásne mesto:-)
Počet znakov v slove je: 25
>>>
```

Index

- Číslo (v hranatých zátvorkách), pomocou ktorého sa vieme dostať ku konkrétnemu znaku reťazca
- indexovanie je od **0** po **dĺžku reťazca - 1**

```
retazec = 'Python'
print(retazec[0]) # vypíše 'P'
print(retazec[1]) # vypíše 'y'
```

P	y	t	h	o	n
0	1	2	3	4	5
-6	-5	-4	-3	-2	-1

```
a = 'Python'
for i in range(len(a)):
        print(i, a[i])
```

```
0 P
1 y
2 t
3 h
4 o
5 n
>>>
```

Funkcia **range(len(a))** zabezpečí, že cyklus prejde postupne pre všetky i od 0 do len(a)-1.

```
-1 n
-2 o
-3 h
-4 t
-5 y
-6 P
>>>
```

```
a = 'Python'
for i in range(1, len(a)+1):
        print(-i, a[-i])
```

Konštrukcia nového reťazca

- Reťazec (string) je v Pythone nemenný typ (immutable), to znamená, že sa nám nepodarí zmeniť znak, napr. retazec[0] = 'J' (Python vtedy ohlási chybu).
- Namiesto zmeny reťazca musíme vždy konštruovať nový reťazec (môže mať aj rovnaké meno).

```
>>> meno = 'Hana'
>>> meno = 'J'+meno[1]+meno[2]+meno[3]
>>> meno
'Jana'
>>>
```

Celočíselné operácie

- oba operandy musia byť celočíselného typu

`+`	sčitovanie	`1 + 2` má hodnotu `3`
`-`	odčitovanie	`2 - 5` má hodnotu `-3`
`*`	násobenie	`3 * 37` má hodnotu `111`
`//`	celočíselné delenie,	`22 // 7` má hodnotu `3`
`%`	zvyšok po delení	`22 % 7` má hodnotu `1`
`**`	umocňovanie	`2 ** 8` má hodnotu `256`

Operácie so znakovými reťazcami

`+`	zreťazenie (spojenie dvoch reťazcov)	`'a' + 'b'` má hodnotu `'ab'`
`*`	viacnásobné zreťazenie reťazca	`3 * 'x'` má hodnotu `'xxx'`

Python na prípady aktualizácie nejakej premennej ponúka špeciálny zápis priraďovacieho príkazu:

a = a+1	a+=1	a = a+b	a+=b
a = a-1	a-=1		
a = a*2	a*=2		
a = a/2	a/=2		
a = a//2	a//=2	celočíselné delenie	
a = a%2	a%=2	zvyšok po celočíselnom delení	
a = a**2	a**=2	druhá mocnina	

Podreťazce

- indexovať môžeme **jeden znak reťazca**

```
retazec = 'Python'
print(retazec[0]) # vypíše 'P'
print(retazec[1]) # vypíše 'y'
```

- ale aj nejaký **podreťazec celého reťazca**

```
retazec = 'Python'
print(retazec[3:6]) # vypíše 'hon'
```

Indexovanie znakov s viacerými indexami - rezy (slice)

reťazec[odkiaľ:pokiaľ]

```
retazec = 'Python'
print(retazec[3:6]) # vypíše 'hon'
```

Znak s indexom pokiaľ sa už nebude nachádzať vo výsledku (index o jeden viac)

P	y	t	h	o	n
0	1	2	3	4	5

Krok indexov

- v reze môžeme určiť aj krok indexov
- krok určuje, o koľko sa budeme v indexoch posúvať
- štandardne je krok 1

```
r = 'Programujeme v Pythone'
print(r[0:10:2]) # vypíše 'Pormj'
print(r[-1:-10:-1]) # vypíše 'enohtyP v'
```

Premenná alebo výraz v reze

- rez okrem konkrétnych čísel môže obsahovať aj premenné, alebo nejaký výraz, ktorého výsledkom je číslo použiteľné v reze

```
s = 'Python'
for i in range (len(s)):
    print(s[:i+1]
```

```
P
Py
Pyt
Pyth
Pytho
Python
```

Príklady

```
s = 'Python'
print(s[3:6]) # vypíše hon
print(s[1:3]) # vypíše yt
print(s[ : ]) # vypíše Python
print(s[ :3]) # vypíše Pyt
print(s[3: ]) # vypíše hon
print(s[5:2]) #
print(s[2:5]) # vypíše tho
```

P	y	t	h	o	n
0	1	2	3	4	5
-6	-5	-4	-3	-2	-1

Príklady

```
s = 'Python'
print(s[-5:-2]) # vypíše yth
print(s[-2:-5])
print(s[-2: ]) # vypíše on
print(s[ :-2]) # vypíše Pyth
print(s[-3: ]) # vypíše hon
```

P	y	t	h	o	n
0	1	2	3	4	5
-6	-5	-4	-3	-2	-1

```
s = 'Python'
print(s[0:6:1]) # vypíše Python
print(s[0:6:2]) # vypíše Pto
print(s[1:6:3]) # vypíše yo
print(s[-1:-6:-1]) # vypíše nohty
print(s[-1:-6:1])  #
print(s[-1:-7:-1]) # vypíše nohtyP
print(s[-6:-1:1]) # vypíše Pytho
print(s[::-1]) # vypíše nohtyP
print(s[::1]) # vypíše Python
```

P	y	t	h	o	n
0	1	2	3	4	5
-6	-5	-4	-3	-2	-1

Operácia in

- operátor príslušnosti
- operácia **in**, zisťuje, či sa zadaný podreťazec nachádza v nejakom konkrétnom reťazci
- jej tvar je: podretazec **in** retazec

```
if 'hon' in 'Python':
    print('je tam')          #True
else:
    print('nie je tam')      #False
```

Operácia in a pravdivostné hodnoty (boolean)

- výsledok vyhodnotenia môžeme dať do premennej
- takáto premenná, je typu **boolean** `'bool'`
- typ boolean má len dve hodnoty **True, False**

```
>>> jetam = 'hon' in 'Python'
>>> jetam
True
>>> type(jetam)
<class 'bool' >
```

Operácia in a pravdivostné hodnoty (boolean)

- výsledok vyhodnotenia môžeme dať do premennej
- takáto premenná, je typu **boolean** `'bool'`
- typ boolean má len dve hodnoty **True, False**

```
>>> x = 5
>>> vysledok = x > 7
>>> vysledok
False
>>> type(vysledok)
<class 'bool' >
```

Reťazcové funkcie

Už poznáme tieto štandardné funkcie:

- **len()** - dĺžka reťazca
- **int()** - prevod reťazca na celé číslo
- **float()** - prevod reťazca na desatinné číslo
- **str()** - prevod čísla (aj ľubovoľnej inej hodnoty) na reťazec
- **ord(), chr()** - prevod do a z unicode

Reťazec, znak, kód

- reťazec je postupnosť znakov uzavretá v apostrofoch ' ' alebo v úvodzovkách " „
- každý znak má svoj číselný kód (tabuľka kódovania znakov)

- @ 64
- 1 49
- A 65
- a 97

- ASCII tabuľka

32	SPC	48	0	64	@	80	P	96	`	112	p
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	'	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	|
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o	127	DEL

Funkcia ord()

- **zistí kód znaku**

```
ord ('a ')     97
ord ('z ')    122
```

32	SPC	48	0	64	@	80	P	96	`	112	p
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	'	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	|
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o	127	DEL

```
čísla:              48 – 57
veľká abeceda:      65 – 106
malá abeceda:       97 - 122
```

Funkcia chr()

- zistí pre dané číslo v tabuľke konkrétny znak

chr(64) @
chr(65) A

32	SPC	48	0	64	@	80	P	96	`	112	p
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	'	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	|
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o	127	DEL

```
for i in range (97, 123):
    print(chr(i))
```

Reťazcové metódy

Je to špeciálny spôsob zápisu volania funkcie:
reťazec.metóda(parametre)

- reťazec.**find(podreťazec)** - Vráti index prvého výskytu podreťazca v reťazci.
- reťazec.**lower()** - Vráti reťazec, v ktorom všetky veľké písmená zamení za malé.
- reťazec.**upper()** - Vráti reťazec, v ktorom všetky malé písmená zamení za veľké.
- reťazec.**replace(podreťazec1, podreťazec2)** - Vráti reťazec, v ktorom nahradí všetky výskyty podreťazec1 za reťazec podreťazec2.

Ďalšie reťazcové metódy

- retazec.**isalnum**() - Vráti True ak reťazec obsahuje len alfanumerické znaky (písmená a číslice) a obsahuje aspoň jeden znak. Inak vráti False.
- retazec.**isalpha**() – Vráti True, ak reťazec obsahuje len alfa znaky (písmená) a obsahuje aspoň jeden znak. Inak vráti False.
- retazec.**isnumeric**()- Vráti True, ak reťazec obsahuje len numerické znaky (číslice) a obsahuje aspoň jeden znak. Inak vráti False.
- retazec.**capitalize**() - Vráti kópiu reťazca, v ktorej je prvý znak prevedený na veľké písmeno a ostatné znaky na malé písmená.
- retazec.**swapcase**() - Vráti kópiu reťazca, v ktorej sú všetky výskyty malých písmen nahradené veľkými písmenami a naopak.
- …

Čo robí tento program?

```
samohlasky = 'aeíou'
s= 'Sedí mucha na stene.'
print(s.find(' '))
print(s[:s.find(' ')])
print(s.lower())
print(s.upper())
print(s.replace('stene', '***'))

for znak in samohlasky:
    s = s.replace(znak,'o')
print(s)
```

Formátovanie reťazcov

```
meno = 'Adam'
print('Ahoj '+ meno +', rád Ťa spoznávam.')
print(f'Ahoj {meno}, rád Ťa spoznávam.')
```

Ahoj Adam, rád Ťa spoznávam.
Ahoj Adam, rád Ťa spoznávam.

Ďakujem za pozornosť!