Práce s řetězci

Jiří Zacpal

KATEDRA INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI

KMI/ZPP1 Základy programování v Pythonu 1

Úkol - řešení

```
n = 36
print("Prvočíselný rozklad čísla" , n , "je:")
i=2
while n>=1:
    while n%i==0:
        print(i, end=" ")
        n//=i
    i+=1
```

Práce s řetězci

Řetězce

- Řetězce jsou tvořeny sekvencí znaků , které jsou v počítači reprezentovány pomocí čísel.
- Python podporuje řetězce kódované podle ASCII tabulky nebo pomocí Unicode.
- Zápis řetězce:

```
s = ’ahoj’
s = "ahoj"
```

- V řetězci lze uvést i escape sekvence:

```
\n, \t, \a, \\, \", \', \0
```

- Jednoprvkový řetězec je znak.

Délka řetězce

- Pro délku řetězce použijeme funkci:

```
len(retezec)
```

  Funkce vrátí počet znaků v řetězci.
- Příklad:

```
>>> len('Python')
6
>>> len('')
```

Operátor indexu

`r[i]`

- Operátor vrátí i-tý znak z řetězce r.
- Příklad:

```
>>> 'Python'[0]
'P'
>>> s = 'Python'
>>> s[0]
'P'
>>> s[5]
'n'
>>> s[6]
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
IndexError: string index out of range
```

Příklad

- S pomocí cyklu for vypíšeme všechny znaky řetězce:

```
string = 'Python'

for i in range(len(string)):
    print(string[i])
```

- Co ale když chceme vytisknout jednotlivé číslice obecně n-ciferného čísla a předem nevíme, kolik je n?

Spojování řetězců

- Pro spojování řetězců se používá operátor +:
  `ret1 + ret2`
- Příklad:

```
>>> 'Py' + 'thon'
'Python‘
```

- Pozor na typ operandu:

```
>>> '1' + 1
TypeError: can only concatenate str (not "int") to str
>>> 1 + '1'
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

Příklad

- Program pro vytvoření řetězce s obráceným pořadím znaků:

```
string = 'Python'
reverse = ''

for i in range(len(string)):
    reverse = string[i] + reverse

print(reverse)
```

Porovnávání řetězců

- Operátory == a != lze použít k porovnávání řetězců.
- Přitom dva řetězce jsou stejné, pokud mají stejnou délku a znaky na odpovídajících indexech se rovnají.
- Příklad:

```
>>> string = 'Python'
>>> string == 'Python'
True
>>> 'p' != 'P'
True
```

- Porovnávání je citlivé na velikost písmen:

```
>>> 'python' == 'Python'
False
```

Příklad

- Následující program otestuje, zda je řetězec palindrom:

```
string = 'kobylamamalybok'
reverse = ''

for i in range(len(string)):
    reverse = string[i] + reverse

is_palindrom = string == reverse
print(is_palindrom)
```

- Lze program napsat i bez konstrukce řetězce reverse?

Příklad

```
string = 'kobylamamalybok'
is_palindrom = True
string_len = len(string)
i = 0
n = string_len // 2

while i < n and is_palindrom:
    if string[i] != string[string_len - 1 - i]:
        is_palindrom = False
    i += 1

print(is_palindrom)
```

Úkol 1

- Pro řetězec s a indexy i a j vraťte podřetězec řetězce s všech znaku s indexem k, kde i < k < j.
- Například pro s rovno 'Python', i rovno 2 a j rovno 5 vraťte 'tho‘.

- Řešení:

```
s="Jazyk Python"
i=4
j=8

if(i<0 or j>len(s) or i>j):
    print("Špatně zadané indexy.")
else:
    podretezec=""
    for z in range(i,j+1):
        podretezec+=s[z]
    print(podretezec)
```

Vztah znaků a čísel

- Číslo odpovídající konkrétnímu znaku v ASCII tabulce je možné zjistit pomocí funkce `ord()`.
- Znak odpovídající konkrétnímu číslu v ASCII tabulce lze zjistit pomocí funkce `chr()`.
- Příklad:

```
print(ord("A")) # zobrazí 65
print(chr(65)) # zobrazí A
```

Příklad

- Program pro převod malých písmen na velká:

```
string = 'python'
upper_case = ''

for i in range(len(string)):
    upper_case += chr(ord(string[i]) - 32)

print(upper_case)
```

Metody pro práci s řetězci

- .upper() převede všechna písmena v řetězci na velká
- .lower() převede všechna písmena v řetězci na malá
- .isalpha() vrací True, pokud řetězec obsahuje pouze písmena, jinak False
- .isdigit() vrací True, pokud řetězec obsahuje pouze číslice, jinak False
- .replace(co, čím) v řetězci nahradí všechny výskyty řetězce co řetězcem čím

Příklad – První písmeno řetězce převede na velká písmeno, ostatní na malá

```
retezec = input("Zadej řetězec: ")
novy_retezec = ""
i = 0

for c in retezec:
    if i > 0:
        novy_retezec += c.lower()
    else:
        novy_retezec += c.upper()
    i += 1

print(novy_retezec)
```

Příklad – První písmeno řetězce převede na velká písmeno, ostatní na malá

```
retezec = input("Zadej řetězec: ")
novy_retezec = retezec[0].upper() + retezec[1:].lower()

print(novy_retezec)
```

Úkol 2

- Vytiskněte ASCII znaky a jejich kódy od 32. do 126. znaku včetně.

- Řešení:

```
for z in range(36,127):
    print(f"Znak {chr(z)} má kód {z}")
```

Bodovaný úkol

1. Vytiskněte všechny slova obsažená v řetězci, kde sousední slova jsou oddělená mezerou. Například pro:

'jablko banán hruška‘

vytiskněte:

jablko
banán
hruška.

2. Odstraňte nadbytečné mezery z řetězce obsahující českou vetu. Například pro retezec

```
' Kobyla  má   malý   bok. '
```

vraťte

```
'Kobyla má malý bok.'
```