Computação I - Python
Aula 5 - Manipulação de Strings

Apresentado por: Bernardo F. Costa

Produção DCC-UFRJ

Metodologia de referência https://doi.org/10.5753/wei.2016.9683

Introdução: help

- Função **help()**: material de consulta
- Disponível na shell
- Consulta pode ser sobre tipo, função ou módulo
- Documentação em inglês
- Fonte primária de consulta para muitos módulos
- Eventualmente, não haverá outra fonte disponível

Strings: funções

- Strings tem muitas funcionalidades
- Função **help()** mostra a documentação completa
- **str.upper()**: nova string com texto em maiúsculo
- **str.lower()**: nova string com texto em minúsculo
- Comparação entre strings diferencia caixa alta de caixa baixa

```
# documentação completa de str
>>> help(str)
>>> help(str.upper)
>>> help(str.lower)
# exemplos
>>> str.upper("abcde")
"ABCDE"
>>> str.lower("Pe de Laranja Lima")
"pe de laranja lima"
>>> "ABC" == "abc"
False
```

Strings: funções

- **str.count()**: conta ocorrências de uma string dentro de outra
- Sintaxe: **str.count(texto,sub,inicio,fim)**
- Busca se dá na região entre *inicio* e *fim*
- Valores default: *inicio* $\leftarrow$ 0, *fim* $\leftarrow$ len(*texto*)

```
# documentação de str
>>> help(str.count)
# exemplos
>>> str.count("macaco come banana","a",2,10)
1
>>> str.count("macaco come banana","a")
5
# sintaxe alternativa
>>> "macaco come banana".count("a")
# outra sintaxe alternativa
>>> s = "macaco come banana"
>>> s.count("a")
```

Strings: funções

- **str.index()**: retorna índice da ocorrências de uma string dentro de outra
- Sintaxe: **str.index(texto,sub,inicio,fim)**
- Busca se dá na região entre *inicio* e *fim*
- Valores default: *inicio* $\leftarrow$ 0, *fim* $\leftarrow$ len(*texto*)
- Erro se não encontrar *sub* em *texto*

```
# documentação de str
>>> help(str.index)
# exemplos
>>> str.index('Mariana','a')
1
>>> str.index('Mariana','a',2)
4
>>> str.index('Mariana','a',5,7)
6
>>> str.index('Mariana','ana')
4
# sintaxe alternativa
>>> "Mariana".index("ana")
# outra sintaxe alternativa
>>> s = "Mariana"
>>> s.index("ana")
# erro de execução
>>> str.index('Mariana','x')
```

Strings: funções

- **str.format()**: retorna uma string com dados a partir de um gabarito
- Sintaxe: **str.format(gabarito,parametro1,parametro2,...)**
- *gabarito* entende {} como entrada de dados
- *gabarito* associa *parametroX* a {} no texto

```
# documentação de str
>>> help(str.format)
# exemplos
>>> str.format('A soma de {} e {} eh {}',2,3,2+3)
"A soma de 2 e 3 eh 5"
# sintaxe alternativa
>>> 'A soma de {} e {} eh {}'.format(2,3,2+3)
# outra sintaxe alternativa
>>> s = 'A soma de {} e {} eh {}'
>>> s.format(2,3,2+3)
```

Strings: funções

- As chaves ({}) podem ser numeradas para repetição ou mudança de ordem
- Nomes podem ser usados dentro das chaves ({})

```
# documentação de str
>>> help(str.format)
# exemplos
>>> str.format('A soma de {1} e {0} eh {2}',2,3,2+3)
"A soma de 3 e 2 eh 5"
>>> str.format('{n1}+{n2}={soma}',n1=2,n2=3,soma=2+3)
"2+3=5"
# sintaxe alternativa
>>> 'A soma de {1} e {0} eh {2}'.format(2,3,2+3)
# outra sintaxe alternativa
>>> s = '{n1}+{n2}={soma}'
>>> s.format(n1=2,n2=3,soma=2+3)
```

Strings: funções

- Formatos numéricos usam letras:
  - inteiro → {:d}
  - notação científica → {:e}
  - ponto fixo → {:f}
  - melhor escolha → {:g}

```
# documentação de str
>>> help(str.format)
# exemplos com inteiros
>>> str.format('{:d} + {:d} = {:d}',2,3,2+3)
"2 + 3 = 5"
# notação científica e ponto fixo
>>> str.format('Notacao cientifica = {:e}',200)
"Notacao cientifica = 2.000000e+02"
>>> str.format('Ponto fixo = {:f}',2)
"Ponto fixo = 2.000000"
```

Strings: funções

- Podemos especificar espaço mínimo ocupado (C) e precisão (P):
  - inteiro → {:**C**d}
  - notação científica → {:**C.P**e}
  - ponto fixo → {:**C.P**f}

```
# documentação de str
>>> help(str.format)
# espaços em branco ou zeros a esquerda e casas decimais
>>> str.format('{:3d}',2)
"  2"
>>> str.format('{:03d}',2)
"002"
>>> str.format('Ponto fixo = {:06.2f}',23)
"Ponto fixo = 023.00"
>>> str.format('Ponto fixo arredondado = {:06.2f}',23.316)
"Ponto fixo arredondado = 023.32"
>>> str.format('Ponto fixo = {:06.2f}',2324.1712)
"Ponto fixo = 2324.17"
```

Revisão

- Strings: texto entre aspas
- Operador [] permite ver partes da string
- Uso de []: indexação, fatiamento, subamostragem
- Concatenação (+) e autoconcatenação múltipla (∗)
- +, ∗ e [] estão presentes em todos iteráveis
- funções **str.upper**, **str.lower**, **str.count**, **str.index** e **str.format**

```
# string
>>> s = "Texto de teste"
# Uso de []
>>> s[0]
"T"
>>> s[:5]
"Texto"
>>> s[::3]
"Ttdtt"
# concatenação e autoconcatenação
múltipla
>>> "abc"+" "+"ghi"
"abc ghi"
>>> 3*"abc"
"abcabcabc"
# recomendamos ler
>>> help(str)
```

Autores

- João C. P. da Silva ▸ Lattes
- Carla Delgado ▸ Lattes
- Ana Luisa Duboc ▸ Lattes

Colaboradores

- Anamaria Martins Moreira ▸ Lattes
- Fabio Mascarenhas ▸ Lattes
- Leonardo de Oliveira Carvalho ▸ Lattes
- Charles Figueiredo de Barros ▸ Lattes
- Fabrício Firmino de Faria ▸ Lattes