MC-102 — Aula 18
Strings

Instituto de Computação – Unicamp

25 de Setembro de 2015

Roteiro

Strings

- Strings em Python são listas imutáveis de caracteres.
- Strings são representados por sequências de caracteres entre aspas simples ' ou entre aspas duplas ''.
- O caracter \ antes de ' ou '' serve para desligar a função de terminar a string.

```
a = "Qwerty de Asdf"
b = 'Querty de Asdf'
c = 'Joe\'s Garage'
d = "Joe's Garage'
e = "Ele falou: \"Saia daqui!!\""
f = 'Ele falou: "Saia daqui!!"'
```

Strings

- Caracteres são representados em Python como uma string de um só caracter! (meio circularmente)

```
>>> a='querty'
>>> a
'querty'
>>> a[2]
'e'
>>> a[-1]
'y'
```

- Embora não faça muito sentido é possível fazer isso:

```
>>> a='querty'
>>> a[2][0][0]
'e'
```

Strings

- Strings são imutáveis:

```
>>> a='qwerty'
>>> a[2]
'e'
>>> a[2]='z'
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'str' object does not support item assignment
>>>
```

- O caracter '\n' pode fazer parte de uma string e ele só causa a mudança de linha no **print**.

```
>>> aa='abc\ndef'
>>> aa
'abc\ndef'
>>> print(aa)
abc
def
```

Strings: operações, funções e métodos

- Operador +: concatena 2 strings, e o operador * repete a concatenação (como em listas).

```
>>> 'qwerty'+'poiuy'
'qwertypoiuy'
>>> 3*'abc'
'abcabcabc'
```

- **slice**: funciona como em listas.

```
>>> a='qwerty'
>>> a[2:4]
'er'
```

- A string vazia é representada como `''`.

Strings como listas (imutáveis)

- Strings podem ser processadas como listas imutáveis (no **for** por exemplo).
- Exercício: inverta uma string.

```
def inverte (s):
    inv = ''
    for x in s:
        inv=x+inv
    return inv
```

Strings: operações, funções e métodos

- A função **input** (já vista) lê e retorna uma string.
- O método **strip** retorna uma string sem os brancos e mudança de linhas no inicio e final de uma string.

```
>>> aa=' \n  abcndef   \n'
>>> aa
' \n  abcndef   \n'
>>> print(aa)

  abcndef

>>> aa.strip()
'abcndef'
```

Strings: operações, funções e métodos

- O operador **in** verifica se uma substring é parte de uma outra string.

```
>>> 'tho' in 'python'
True
>>> 'thor' in 'python'
False
```

- O método **find** retorna onde a substring começa na string.

```
>>> p='python'
>>> p.find('tho')
2
>>> p.find('thor')
-1
```

Strings: operações, funções e métodos

- O método **split(sep)** separa uma string usando **sep** como separador. Retorna uma lista das substrings.

```
>>> a="1; 2   ;  3"
>>> a.split(';')
['1', ' 2   ', '  3']
```

- O método **split()** separa usando espaço '\n' e tab como **sep**.

```
>>> b="ouviram   do  ipiranga margens"
>>> b.split()
['ouviram', 'do', 'ipiranga', 'margens']
```

- Note que pode haver substrings vazias.

```
>>> a="1;2;;3"
>>> a.split(';')
['1', '2', '', '3']
```

Strings: operações, funções e métodos

- O método **replace** serve para trocar todas as ocorrências de uma substring por outra em uma string.

```
>>> a="abcabcdfgabc abc a b c"
>>> a.replace("abc","")
'dfg  a b c'
```

- Podemos usar a função **list** para transformar uma string em uma lista onde os itens da lista correspondem aos caracteres da string.

```
>>> a="abc\n;abc"
>>> list(a)
['a', 'b', 'c', '\n', ';', 'a', 'b', 'c']
```

- O método **join** recebe como parâmetro uma sequência ou lista e retorna uma string com a concatenação dos elementos da sequencia/lista.

```
>>> a="abc\n;abc"
>>> l = list(a)
>>> l
['a', 'b', 'c', '\n', ';', 'a', 'b', 'c']
>>> "".join(l)
'abc\n;abc'
```

Processamento de Texto

- Como exemplo de funções com strings vamos implementar duas funcionalidades básicas de processadores de texto:
  1. Contar o número de palavras em um texto.
  2. Fazer a busca de todas as ocorrências de uma palavra em um texto.

Processamento de Texto

Função para contar o número de palavras em uma string.

```
def numPalavras(st):
    st = delPontuacao(st)
    return len(st.split())

def delPontuacao(st): #devolve a string sem as pontuações
    pontuacao = [".", ",", ":", ";", "!", "?"]
    for i in pontuacao:
        st = st.replace(i, "")
    return st

st = "ola turma de mc102!\nAqui jaz um incompreendido, um torto, um heroi!"
print("Num palavras:", numPalavras(st))
```

Processamento de Texto

Achar palavras em um texto.

- Fazer função que acha todas as posições de ocorrência de uma palavra (substring) em um texto (uma string).
- **def findAll(subst, st):**

Exemplo:

```
Texto=a tete tetete
Palavra=tete

A resposta é [2, 7, 9]
```

Processamento de Texto

Ideia do algoritmo:

- Achamos a posição de ocorrência de **subst** em **st** e armazenamos esta na lista **pos**.
- Depois removemos toda parte inicial de **st** até o primeiro caractere onde encontramos **subst**:
  Exe: subst="abc" e st="dfg abcabc", vamos remover "dfg a" ficando st="bcabc".
- Como removemos uma parte inicial de **st** precisamos guardar o seu tamanho em **tamRemovido** pois próximas ocorrências estão deslocadas por este valor na string original.

```
def findAll(subst, st):
    pos = []
    tamRemovido = 0
    while subst in st:
        aux = st.find(subst) #acha posicao da 1o ocorrência
        pos.append(aux+tamRemovido)
        st = st[(aux+1):] #remove tudo até 1a letra da 1o ocorrência de subst
        tamRemovido = tamRemovido + aux + 1
    return pos
```

Processamento de Texto

```
def findAll(subst, st):
    pos = []
    tamRemovido = 0
    while subst in st:
        aux = st.find(subst) #acha posicao da 1o ocorrência
        pos.append(aux+tamRemovido)
        st = st[(aux+1):] #remove tudo até 1a letra da 1o ocorrência de subst
        tamRemovido = tamRemovido + aux + 1
    return pos

print(findAll("abc", "abc , jkdf abcabc \na b c"))
print(findAll("a", "baaaaa"))
print(findAll("a", ""))
print(findAll("tete", "a tete tetete"))
```

Exercício

- Escreva um programa que lê uma string, e imprime "Palindromo" caso a string seja um palindromo e "Nao Palindromo" caso contrário.
- OBS: Um palindromo é uma palavra ou frase, que é igual quando lida da esquerda para a direita ou da direita para a esquerda (espaços em brancos são descartados). Assuma que a entrada não tem acentos e que todas as letras são minúsculas
- Exemplo de palindromo: "saudavel leva duas"
- Faça uma nova versão que aceita como palindromo mesmo que as letras correspondentes sejam maiúsculas e minúsculas. Assim "Saudavel Leva DUas" deve ser também um palindromo.

Exercício

- O usuário entra cinco números separados por brancos. Imprima a média deles.
- O usuário entra com vários números separados por branco ou virgula, por exemplo “3,4 5 6, 9” . Imprima a média deles.