MC102 - Algoritmos e Programação de Computadores

Turma Z - Segundo Semestre de 2019

A partir desse slide, utilizaremos o material desenvolvido pela professora Sandra Avila e disponível em http://www.ic.unicamp.br/~sandra/

Agenda

- Strings
  - Operações
  - Funções
  - Métodos
- Exercícios

Strings

- Strings em Python são listas **imutáveis** de caracteres.
- Strings são representadas por sequências de caracteres entre aspas simples `'` ou entre aspas duplas `"`.

```
a = "26 de Abril tem prova."
a
'26 de Abril tem prova.'
b = 'Fizeram a atividade conceitual?'
b
'Fizeram a atividade conceitual?'
c = "Que vida \"fácil\""
c
'Que vida "fácil"'
```

Strings

- Strings em Python são listas **imutáveis**, portanto pode-se acessar posições de uma string de forma usual.

```
a = "26 de Abril tem prova."
a[0]
'2'
a[0] = "1"
```

```
---------------------------------------------------------------
TypeError                 Traceback (most recent call last)
<ipython-input-13-9ab1dda42293> in <module>()
----> 1 a[0] = "1"

TypeError: 'str' object does not support item assignment
```

Strings

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

```
a = 'Fizeram\na\natividade\nconceitual?'
a
'Fizeram\na\natividade\nconceitual?'
```

```
a = 'Fizeram\na\natividade\nconceitual?'
print(a)
Fizeram
a
atividade
conceitual?
```

Strings: Operações, Funções e Métodos

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

```
a = "26 de Abril tem prova."
b = 'Fizeram a atividade conceitual?'
a + b
'26 de Abril tem prova.Fizeram a atividade conceitual?'
```

```
b = 'Fizeram a atividade conceitual?\n'
print(3*b)
Fizeram a atividade conceitual?
Fizeram a atividade conceitual?
Fizeram a atividade conceitual?
```

Strings como Listas

- Strings podem ser processadas como listas, podendo por exemplo ter seus elementos percorridos num laço **for**.
- Exemplo: Ler uma string e imprimir a inversa.

```
string = input("Digite um texto: ")
inversa = " "
for x in string:
    inversa = x + inversa
print(inversa)
```

Strings: Operações, Funções e Métodos

- A função **slice** (fatiar) devolve a string entre duas posições dadas.
- Pode-se fatiar (slice) strings usando `[inicio:fim-1:passo]`.

```
a = "20 de Abril tem prova."
a[6:11]
'Abril'
a[6:11:2]
'Arl'
a[::-1]
'.avorp met lirbA ed 02'
```

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

Strings: Operações, Funções e Métodos

- O método `strip` retorna uma string sem os brancos e mudança de linhas **no início e no final** de uma string.

```
b = "Fizeram a atividade conceitual?"
b
'\n Fizeram a atividade conceitual? \n'

b.strip()
'Fizeram a atividade conceitual?'
```

Strings: Operações, Funções e Métodos

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

```
"atividade" in "Fizeram a atividade conceitual?"
True

"idade" in "Fizeram a atividade conceitual?"
True

"Abril" in "Fizeram a atividade conceitual?"
False
```

Strings: Operações, Funções e Métodos

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

```
a = "Fizeram a atividade conceitual?"
a.find("atividade")
10

a.find("abril")
-1
```

- O método `find` retorna `-1` quando a substring não ocorre na string.

Strings: Operações, Funções e Métodos

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

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

a = "Fizeram a atividade conceitual?"
a.split()
['Fizeram', 'a', 'atividade', 'conceitual?']
```

- Podem haver substrings vazias no retorno de `split()`.

Strings: Operações, Funções e Métodos

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

```
a = "Fizeram a atividade conceitual?"
a.replace("conceitual","teórica")
'Fizeram a atividade teórica?'

a = "Fizeram a atividade conceitual?"
a.replace("conceitual","")
'Fizeram a atividade ?'
```

Strings: Operações, Funções e Métodos

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

```
numeros = "1; 2 ;  3"
list(numeros)
['1', ';', ' ', '2', ' ', ';', ' ', ' ', '3']

list("atividade")
['a', 't', 'i', 'v', 'i', 'd', 'a', 'd', 'e']
```

Strings: Operações, Funções e Métodos

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

```
l = list("atividade")
l
['a', 't', 'i', 'v', 'i', 'd', 'a', 'd', 'e']

"".join(l)
'atividade'
```

Exercícios

Exemplo: Contador de Palavras

- Faça um programa que conta o número de palavras em um texto.

Exemplo: Contador de Palavras

- Faça um programa que conta o número de palavras em um texto.
  - Primeiramente removemos do texto todos os sinais de pontuação.

```
texto = input("Digite um texto: ")
pontuacao = [".", ",", ":", ";", "!", "?"]

# remove os sinais de pontuação
for p in pontuacao:
    texto = texto.replace(p," ")
```

Exemplo: Contador de Palavras

- Faça um programa que conta o número de palavras em um texto.
  - Depois usamos a função `split` para separar as palavras.

```
texto = input("Digite um texto: ")
pontuacao = [".", ",", ":", ";", "!", "?"]

# remove os sinais de pontuação
for p in pontuacao:
    texto = texto.replace(p," ")

# split devolve lista com palavras como itens
numero_palavras = len(texto.split())
print("Número de palavras:", numero_palavras)
```

Exercício: Palíndromo

- Faça um programa que lê uma string e imprime “Palíndromo” caso a string seja um palíndromo e “Não é palíndromo” caso não seja.
    - Assuma que a entrada não tem acentos e que todas as letras são minúsculas.

- Obs: Um *palíndromo* 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).
    - Exemplos de palíndromo: “ovo”, “reviver”, “mega bobagem”, “anotaram a data da maratona”

Exercício: Palíndromo

- Faça um programa que lê uma string e imprime “Palíndromo” caso a string seja um palíndromo e “Não é palíndromo” caso não seja.

Entrada	Saída
ovo	Palíndromo

Entrada	Saída
prova	Não é palíndromo

Entrada	Saída
anotaram a data da maratona	Palíndromo

Exercício: Palíndromo

```
texto = input("Digite um texto: ")

# inverte a string
texto_inverso = texto[::-1]

# remove os espaços em branco
texto = texto.replace(" ","")
texto_inverso = texto_inverso.replace(" ","")

# verifica se texto é igual ao texto_inverso
if (texto == texto_inverso):
    print("Palíndromo")
else:
    print("Não é palíndromo")
```

Exercício: Palíndromo

```
texto = input("Digite um texto: ")

# remove os espaços em branco
texto = texto.replace(" ","")
texto_inverso = texto_inverso.replace(" ","")

# inverte a string
texto_inverso = texto[::-1]

# verifica se texto é igual ao texto_inverso
if (texto == texto_inverso):
    print("Palíndromo")
else:
    print("Não é palíndromo")
```

Exercício: Palíndromo

- Faça uma nova versão que aceita como palíndromo mesmo que as letras correspondentes sejam maiúsculas e minúsculas.
  - Exemplo: “Ovo”, “Anotaram a Data da Maratona” devem ser também palíndromo.

Exercício: Palíndromo

```
texto = input("Digite um texto: ")

# inverte a string
texto_inverso = texto[::-1]

# remove os espaços em branco
texto = texto.replace(" ","")
texto_inverso = texto_inverso.replace(" ","")

# verifica se texto é igual ao texto_inverso
if (texto.lower() == texto_inverso.lower()):
    print("Palíndromo")
else:
    print("Não é palíndromo")
```

Exercício: Palíndromo

```
texto = input("Digite um texto: ")

# inverte a string
texto_inverso = texto[::-1]

# remove os espaços em branco
texto = texto.replace(" ","")
texto_inverso = texto_inverso.replace(" ","")

# verifica se texto é igual ao texto_inverso
if (texto.lower() == texto_inverso.lower()):
    print("Palíndromo")
else:
    print("Não é palíndromo")
```

Strings: Resumo

Método	Parâmetros	Descrição
`strip`	nenhum	Retorna uma string removendo caracteres em branco do início e do fim. Ex: `a.strip()`
`find`	substring	Retorna o índice onde a substring começa na string. Ex: `a.find("texto")`
`split`	nenhum	Separa uma string usando sep como separador e retorna uma lista das substrings. Ex: `a.split()`
`replace`	substring1, substring2	Substitui todas as ocorrências de uma substring por outra. Ex: `a.replace("prova","teste")`
`list`	substring	Transforma uma string em uma lista onde os itens da lista correspondem aos caracteres da string. Ex: `list("texto")` ou `list(a)`
`join`	substring	Retorna uma string com a concatenação dos elementos da sequência/lista. Ex: `"".join(a)`
`count`	substring	Retorna o número de ocorrências de uma substring. Ex: `a.count("as")`
`upper`	nenhum	Retorna uma string toda em maiúsculas. Ex: `a.upper()`
`lower`	nenhum	Retorna uma string toda em minúsculas. Ex: `a.lower()`

Exemplos & Exercícios

Exemplo: Conta espaços e vogais

- Faça um programa que conta espaços e vogais. Dado um texto (sem acento) informado pelo usuário, conte:
  - Quantos espaços em branco existem no texto.
  - Quantas vezes aparecem as vogais a, e, i, o, u.

Entrada	Saída
`24 de Abril tem revisao para a prova`	`espaços: 7` `a: 6 e: 3 i: 2 o: 2 u: 0`

Solução 1

Exemplo: Conta espaços e vogais

```
texto = input("Digite um texto: ")

# conta espaços em branco
numero_espacos = texto.count(" ")
print("espaços:", numero_espacos)

# conta vogais
vogal_a = texto.lower().count("a")
vogal_e = texto.lower().count("e")
vogal_i = texto.lower().count("i")
vogal_o = texto.lower().count("o")
vogal_u = texto.lower().count("u")
print("a:", vogal_a, "e:", vogal_e, "i:", vogal_i, "o:",
      vogal_o, "u:", vogal_u)
```

Solução 2

Exemplo: Conta espaços e vogais

```
texto = input("Digite um texto: ")

# conta espaços em branco
numero_espacos = texto.count(" ")
print("espaços:", numero_espacos)

# conta vogais
vogais = ["a", "e", "i", "o", "u"]
for v in vogais:
    numero_vogais = texto.lower().count(v)
    print(str(v) + ": " + str(numero_vogais), end=" ")
```

Solução 3

Exemplo: Conta espaços e vogais

```
texto = input("Digite um texto: ")

# conta espaços em branco
numero_espacos = texto.count(" ")
print("espaços:", texto.count(" "))

# conta vogais
vogais = ["a", "e", "i", "o", "u"]
for v in vogais:
    numero_vogais = texto.lower().count(v)
    print(str(v) + ": " + str(texto.lower().count(v)), end=" ")
```

Solução 3

Exemplo: Conta espaços e vogais

```
texto = input("Digite um texto: ")

# conta espaços em branco
print("espaços:", texto.count(" "))

# conta vogais
vogais = ["a", "e", "i", "o", "u"]
for v in vogais:
    print(str(v) + ": " + str(texto.lower().count(v)), end=" ")
```

Solução 3.1

Exemplo: Conta espaços e vogais

```
texto = input("Digite um texto: ")

# conta espaços em branco
print("espaços:", texto.count(" "))

# conta vogais
vogais = ["a", "e", "i", "o", "u"]
for v in vogais:
    print(v, ":", texto.lower().count(v), end=" ")
```

Exercício: Data por extenso

- Faça um programa que solicite a data de nascimento (dd/mm/aaaa) do usuário e imprima a data com o nome do mês por extenso.

Entrada	Saída
`16/12/1982`	`16 de dezembro de 1982`

Exercício: Crime

- Utilizando listas, faça um programa que faça 5 perguntas para uma pessoa sobre um crime. As perguntas são:
  - “Telefonou para a vítima?”
  - “Esteve no local do crime?”
  - “Mora perto da vítima?”
  - “Devia para a vítima?”
  - “Já trabalhou com a vítima?”
- Se a pessoa responder positivamente a 2 questões ela deve ser classificada como “Suspeita”, entre 3 e 4 como “Cúmplice” e 5 como “Assassino”. Caso contrário, ele será classificado como “Inocente”.

```
res = []
res.append(input("Telefonou para a vítima? 1/Sim ou 0/Não: "))
res.append(input("Esteve no local do crime? 1/Sim ou 0/Não: "))
res.append(input("Mora perto da vítima? 1/Sim ou 0/Não: "))
res.append(input("Devia para a vítima? 1/Sim ou 0/Não: "))
res.append(input("Já trabalhou com a vítima? 1/Sim ou 0/Não: "))
```

Complete o programa ...

Solução 1

```
lista_perguntas = ["Telefonou para a vítima? 1/Sim ou 0/Não: ",
                   "Esteve no local do crime? 1/Sim ou 0/Não: ",
                   "Mora perto da vítima? 1/Sim ou 0/Não: ",
                   "Devia para a vítima? 1/Sim ou 0/Não: ",
                   "Já trabalhou com a vítima? 1/Sim ou 0/Não: "]
```

Complete o programa ...

Solução 2

```
# Essa solução não utiliza listas
res1 = int(input("Telefonou para a vítima? 1/Sim ou 0/Não: "))
res2 = int(input("Esteve no local do crime? 1/Sim ou 0/Não: "))
res3 = int(input("Mora perto da vítima? 1/Sim ou 0/Não: "))
res4 = int(input("Devia para a vítima? 1/Sim ou 0/Não: "))
res5 = int(input("Já trabalhou com a vítima? 1/Sim ou 0/Não: "))
```

Complete o programa ...

Solução 3

Desafio: Jogo da Forca

- Faça um jogo da forca. O programa terá uma lista de palavras lidas de um arquivo texto e escolherá uma aleatoriamente. O jogador poderá errar 6 vezes antes de ser enforcado.

```
Digite uma letra: a
-> Você errou pela 1a vez. Tente de novo!

Digite uma letra: o
A palavra é: _ _ _ _ o

Digite uma letra: e
A palavra é: _ e _ _ o

Digite uma letra: s
-> Você errou pela 2a vez. Tente de novo!
```

```
import random # importa o módulo random
palavras = input("Digite as palavras: ")
palavras = palavras.split(" ")

# pega um número aleatoriamente entre 0 e número de palavras
uma_palavra = palavras[random.randrange(0,len(palavras))]
```

Complete o programa ...

Referências & Exercícios

- https://wiki.python.org.br/ExerciciosComStrings: 14 exercícios =)
- https://wiki.python.org.br/ExerciciosListas: 24 exercícios =)

- https://panda.ime.usp.br/pensepy/static/pensepy/08-Strings/strings.html
- https://panda.ime.usp.br/pensepy/static/pensepy/09-Listas/listas.html