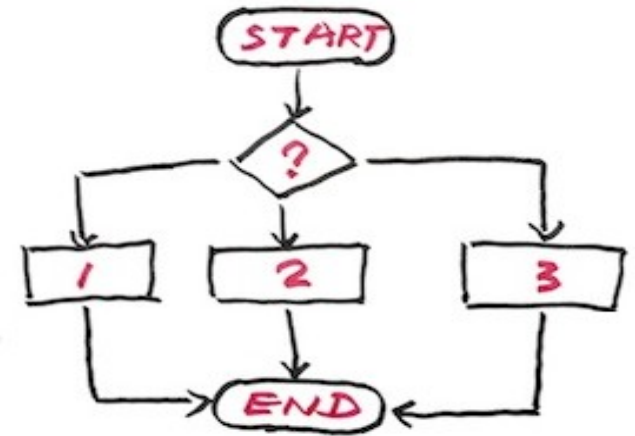


Introdução à Programação

Aula 11 *String*

Prof. Dr. Ivan José dos Reis Filho



Aula anterior

- Switch

Aula de hoje

Introdução às Strings

- Fatiando
- Modificando
- Concatenando
- Format
- Escape
- Métodos

Introdução

String é qualquer sequência de caracteres, classificados em 3 grupos:

- Numérico;
- Letras do alfabeto;
- Caracteres especiais.

```
a = "Hello"  
b = "10"  
  
print(a)  
print(int(b))
```

Em python, todo texto ou caractere é um objeto do tipo str. Caso necessário, precisa-se fazer a conversão.

Conteúdo

Strings

- **Fatiando**
- Modificando
- Concatenando
- Format
- Escape
- Métodos

Fatiamento

Slicing

Podemos retornar um intervalo de caracteres usando a sintaxe de fatiar.

- Necessário especificar o índice inicial e final;
- Separar por dois pontos.

```
b = "Hello, World!"  
print(b[2:5])
```

llo

Fatiamento

Slicing

Podemos retornar um intervalo de caracteres usando a sintaxe de fatiar.

- Fatiar a partir do início.
- Por deixar a **primeira posição** em branco, o intervalo inicial é a primeira posição.

```
b = "Hello, World!"  
print(b[:5])
```

Hello

Fatiamento

Slicing

Podemos retornar um intervalo de caracteres usando a sintaxe de fatiar.

- Fatiar a partir do início.
- Por deixar a **última posição** posição em branco, o intervalo termina vai até o último caracter.

```
b = "Hello, World!"  
print(b[2:])
```

```
llo, World!
```


Fatiamento

Slicing

Podemos retornar um intervalo de caracteres usando a sintaxe de fatiar.

- Índices negativos:
- Use o índice negativo para começar o fatiamento a partir da última posição da string.

```
b = "Hello, World!"  
print(b[-5:-2])
```

```
orl
```

Atividade 01

Faça uma algoritmo para receber o nome completo do usuário. Faça as seguintes operações

- a) Mostre apenas os 10 primeiros caracteres
- b) Mostre apenas os 10 últimos caracteres
- c) Mostre a partir do terceiro caracter até o antenúltimo caracter.

Conteúdo

Strings

- Fatiando
- **Modificando**
- Concatenando
- Format
- Escape
- Métodos

Modificando

Modify

Python tem um conjunto de métodos “built-in”

- **Upper Case:**
- Retorna a string em CAIXA ALTA.

```
a = "Hello, World!"  
print(a.upper())
```

```
HELLO, WORLD!
```

Modificando

Modify

Python tem um conjunto de métodos “built-in”

- **Lower Case:**
- Retorna a string em caixa baixa.

```
a = "Hello, World!"  
print(a.lower())
```

```
hello, world!
```

Modificando

Modify

Python tem um conjunto de métodos “built-in”

- **strip()**:
- Remove qualquer espaço em branco no começo ou no final

```
a = " Hello, World! "  
print(a.strip())
```

```
Hello, World!
```

Modificando

Modify

Python tem um conjunto de métodos “built-in”

- **replace()**:
- Substitue uma string/caracter com outra string/caracter

```
a = "Hello, World!"  
print(a.replace("H", "J"))
```

```
Jello, World!
```

Modificando

Modify

Python tem um conjunto de métodos “built-in”

- **split()**:
- retorna uma lista em que o texto entre o separador especificado se torna os itens da lista.

```
a = "Hello, World!"  
  
b = a.split(",")  
print(b)
```

```
['Hello', ' World!']
```


Modificando

Modify

Veja mais sobre os métodos Strings.

https://www.w3schools.com/python/python_ref_string.asp

Method

capitalize()

casefold()

center()

count()

encode()

endswith()

expandtabs()

find()

format()

format_map()

index()

isalnum()

isalpha()

isascii()

isdecimal()

isdigit()

isidentifier()

islower()

isnumeric()

isprintable()

isspace()

istitle()

isupper()

join()

ljust()

lower()

lstrip()

maketrans()

partition()

Atividade 02

Faça um algoritmo que **recebe** o nome completo do usuário. Faça as seguintes operações

- a) Transforme todas letras em maiúsculas
- b) Transforme todas letras em minúsculas
- c) Substitua as vogais 'a' por 'i'
- d) Separe o nome e sobrenomes.

Method

capitalize()

casefold()

center()

count()

encode()

endswith()

expandtabs()

find()

format()

format_map()

index()

isalnum()

isalpha()

isascii()

isdecimal()

isdigit()

isidentifier()

islower()

isnumeric()

isprintable()

isspace()

istitle()

isupper()

join()

ljust()

lower()

lstrip()

maketrans()

partition()

Conteúdo

Strings

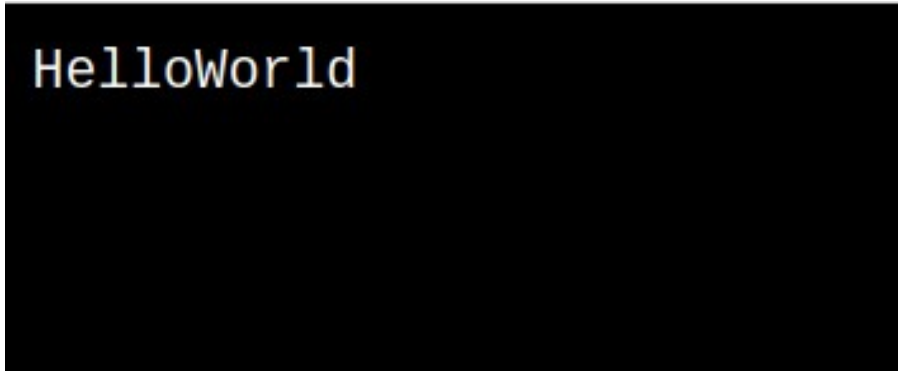
- Fatiando
- Modificando
- **Concatenando**
- Format
- Escape
- Métodos

Strings

Concatenação

Concatenar é a junção de duas cadeiras de caracteres;

```
a = "Hello"  
b = "World"  
  
c = a + b  
  
print(c)
```



HelloWorld

A concatenação, pode ser a direita ou então a esquerda de outra string

Strings

Concatenação

Para adicionar espaços entre as strings

```
a = "Hello"  
b = "World"  
  
c = a + " " + b  
  
print(c)
```

```
Hello World
```

Conteúdo

Strings

- Fatiando
- Modificando
- Concatenando
- **Format**
- Escape
- Métodos

Strings

Interpolação: format

Não podemos combinar strings e números (variável)

```
age = 36  
txt = "My name is John, I am " + age  
print(txt)
```

```
Traceback (most recent call last):  
  File "demo_string_format_error.py", line 2, in <module>  
    txt = "My name is John, I am " + age  
TypeError: must be str, not int
```

Strings

Interpolação

Exemplo:

```
age = 36  
txt = "My name is John, and I am {}"  
print(txt.format(age))
```

```
My name is John, and I am 36
```


Strings

Interpolação

Exemplo:

```
quantity = 3  
itemno = 567  
price = 49.95  
myorder = "I want {} pieces of item {} for {} dollars."  
print(myorder.format(quantity, itemno, price))
```

```
I want 3 pieces of item 567 for 49.95 dollars.
```

Conteúdo

Strings

- Fatiando
- Modificando
- Concatenando
- Format
- **Escape**
- Métodos

Strings

escape

Para inserir caracteres ilegais em uma string, use um caractere de escape.

- Um caractere de escape é uma **barra invertida ** seguida do **caractere que você deseja inserir**.
- Um **exemplo** de caractere ilegal é uma aspa dupla dentro de uma string entre aspas duplas:

Você receberá um erro se usar aspas duplas dentro de uma string entre aspas duplas:

```
txt = "We are the so-called "Vikings" from the north."
```

Strings

escape

Para corrigir o problema, use o caractere de escape \"

– Exemplo:

```
txt = "We are the so-called \"Vikings\" from the north."  
print(txt)
```

```
We are the so-called "Vikings" from the north.
```

Strings

escape

Outros caracteres

Code	Result
\'	Single Quote
\\	Backslash
\n	New Line
\r	Carriage Return
\t	Tab
\b	Backspace
\f	Form Feed
\ooo	Octal value
\xhh	Hex value

Conteúdo

Strings

- Fatiando
- Modificando
- Concatenando
- Format
- Escape
- **Métodos**

Strings

funções

Method	Description
<u><a>capitalize()</u>	Converts the first character to upper case
<u><a>casefold()</u>	Converts string into lower case
<u><a>center()</u>	Returns a centered string
<u><a>count()</u>	Returns the number of times a specified value occurs in a string
<u><a>encode()</u>	Returns an encoded version of the string
<u><a>endswith()</u>	Returns true if the string ends with the specified value
<u><a>expandtabs()</u>	Sets the tab size of the string
<u><a>find()</u>	Searches the string for a specified value and returns the position of where it was found
<u><a>format()</u>	Formats specified values in a string

Strings

funções

<code>format_map()</code>	Formats specified values in a string
<code><u>index()</u></code>	Searches the string for a specified value and returns the position of where it was found
<code><u>isalnum()</u></code>	Returns True if all characters in the string are alphanumeric
<code><u>isalpha()</u></code>	Returns True if all characters in the string are in the alphabet
<code><u>isdecimal()</u></code>	Returns True if all characters in the string are decimals
<code><u>isdigit()</u></code>	Returns True if all characters in the string are digits
<code><u>isidentifier()</u></code>	Returns True if the string is an identifier
<code><u>islower()</u></code>	Returns True if all characters in the string are lower case
<code><u>isnumeric()</u></code>	Returns True if all characters in the string are numeric
<code><u>isprintable()</u></code>	Returns True if all characters in the string are printable
<code><u>isspace()</u></code>	Returns True if all characters in the string are whitespaces
<code><u>istitle()</u></code>	Returns True if the string follows the rules of a title
<code><u>isupper()</u></code>	Returns True if all characters in the string are upper case

Strings

funções

<u>join()</u>	Joins the elements of an iterable to the end of the string
<u>ljust()</u>	Returns a left justified version of the string
<u>lower()</u>	Converts a string into lower case
<u>lstrip()</u>	Returns a left trim version of the string
<u>maketrans()</u>	Returns a translation table to be used in translations
<u>partition()</u>	Returns a tuple where the string is parted into three parts
<u>replace()</u>	Returns a string where a specified value is replaced with a specified value
<u>rfind()</u>	Searches the string for a specified value and returns the last position of where it was found
<u>rindex()</u>	Searches the string for a specified value and returns the last position of where it was found
<u>rjust()</u>	Returns a right justified version of the string
<u>rpartition()</u>	Returns a tuple where the string is parted into three parts
<u>rsplit()</u>	Splits the string at the specified separator, and returns a list
<u>rstrip()</u>	Returns a right trim version of the string
<u>split()</u>	Splits the string at the specified separator, and returns a list

Strings

funções

<u>split()</u>	Splits the string at the specified separator, and returns a list
<u>splitlines()</u>	Splits the string at line breaks and returns a list
<u>startswith()</u>	Returns true if the string starts with the specified value
<u>strip()</u>	Returns a trimmed version of the string
<u>swapcase()</u>	Swaps cases, lower case becomes upper case and vice versa
<u>title()</u>	Converts the first character of each word to upper case
<u>translate()</u>	Returns a translated string
<u>upper()</u>	Converts a string into upper case
<u>zfill()</u>	Fills the string with a specified number of 0 values at the beginning

Exercícios

Exercícios

1. Dada uma string, imprima o seu comprimento.
2. Dada uma string, imprima apenas o primeiro e o último caractere.
3. Dada uma string, imprima os caracteres nos índices pares.
4. Dada uma string, verifique se ela é um palíndromo (ou seja, se pode ser lida da mesma forma da esquerda para a direita e vice-versa).
5. Dada uma string, substitua todas as ocorrências de um caractere específico por outro caractere.

Exercícios

6. Dada uma string, concatene-a com outra string fornecida pelo usuário.
7. Dada uma string, inverta a ordem dos caracteres.
8. Dada uma string, conte quantas vezes um determinado caractere aparece nela.
9. Dada uma string, verifique se ela começa ou termina com uma determinada sequência de caracteres.
10. Dada uma string contendo uma frase, capitalize a primeira letra de cada palavra.