

VERIFICAÇÃO DE STRINGS

- `startswith` e `endswith` para validar se a string começa ou termina com uma determinada sequência de caracteres.
- Podemos também verificar se uma palavra está contida na String usando o operador `in`.
- Podemos também verificar se uma palavra não está contida na String usando `not in`

```
cidade = 'São Carlos'
endereco = 'Rua Cândido Padim, 25 - Vila Prado'
completo = cidade + endereco
print(cidade.startswith('São'))
print(cidade.endswith('los'))
print('Rua' in completo)
print('Avenida' not in completo)
```

CONTANDO E PESQUISANDO ELEMENTOS EM UMA STRING

- Podemos contar elementos em uma string usando o método **count**
- Podemos utilizar o método **find** para obter a posição da primeira ocorrência de uma String e podemos utilizar o método **rfind** para realizar a pesquisa da direita para a esquerda
- Tanto **find** quanto **rfind** possuem dois parâmetros para que sejam delimitados o início e o fim para realização da pesquisa
- Para localizar o índice de ocorrência de uma string, podemos usar também o **index** e **rindex**, porém, caso a substring não seja localizada, esses métodos geram uma exceção **ValueError**

EXEMPLO

```
texto = 'Python é uma linguagem de programação. Python é simples. Python é  
organizado. Python é uma excelente linguagem.'  
print(texto.count('é'))  
print(texto.find('Python',25,50))  
print(texto.rfind('lingua'))  
print(texto.index('é'))  
print(texto.rindex('é'))
```

POSICIONAMENTO DE STRINGS

- O método **center** centraliza uma string em um número de posições passado como parâmetro, preenchendo com espaços à direita e à esquerda, até que a string esteja centralizada. Podemos substituir os espaços por qualquer caractere informando como parâmetro para a função.
- O método **ljust** alinha o texto à esquerda e preenche com espaços à direita, já o método **rjust** alinha o texto à direita e preenche com espaços à esquerda

EXEMPLO

```
texto = 'Olá Mundo!'
texto_centro = texto.center(20)
texto_centro_2 = texto.center(20, '=')
texto_esquerda = texto.ljust(12)
texto_direita = texto.rjust(12)
print(f'**{texto_esquerda}*{texto_direita}**')
```

Olá Mundo! * Olá Mundo!

12 Espaços

SEPARAÇÃO DE STRINGS

- Para separar strings à partir de um determinado caractere podemos utilizar o método **split**. O resultado é uma lista com os itens que eram delimitados pelo caractere informado na chamada ao método **split**

```
nomes = "João Paulo/Maria Paula/Ana Beatriz/José Pedro"  
print(nomes.split('/'))
```

- Para separar strings à partir do caractere de nova linha “\n” podemos usar o método **splitlines**.

```
nomes = "João Paulo\nMaria Paula\nAna Beatriz\nJosé Pedro"  
print(nomes.splitlines())
```

SUBSTITUIÇÃO DE STRINGS E REMOÇÃO DE ESPAÇOS EM BRANCO

- Para substituir trechos de uma string podemos usar o método **replace**. Este método recebe por parâmetro a string que será substituída, a nova string, e por fim um número que limitará a quantidade de substituições.
- Se o primeiro parâmetro do **replace** for vazio, será inserido o caractere informado antes de cada caractere da string e se o segundo parâmetro for vazio, o trecho será apagado

EXEMPLO

```
f = 'A força eletromotriz induzida em qualquer circuito fechado é igual ao negativo  
da variação do fluxo magnético com o tempo na área delimitada pelo circuito'  
f1 = f.replace('força', 'Bicicleta')  
f2 = f.replace(", '#")  
print(f1)  
print(f2)
```


REMOÇÃO DE ESPAÇOS

```
a = '  Olá mundo  '
print(f'*{a}*')
b = a.strip()
print(f'*{b}*')
c = a.lstrip()
print(f'*{c}*')
d = a.rstrip()
print(f'*{d}*')
```

- Para remover espaços em branco de uma string temos três métodos:
- Strip: Remove espaços no início e fim da string.
- Lstrip: Remove espaços no início da string.
- Rstrip: Remove espaços no fim da string.

EXERCÍCIOS

1. Faça um programa que leia 2 strings e informe o conteúdo delas seguido do seu comprimento. Informe também se as duas strings possuem o mesmo comprimento e são iguais ou diferentes no conteúdo.
2. Faça um programa que solicite o nome do usuário e imprima-o na vertical.
3. Faça um programa que permita ao usuário digitar o seu nome e em seguida mostre o nome do usuário de trás para frente utilizando somente letras maiúsculas. Dica: lembre-se que ao informar o nome o usuário pode digitar letras maiúsculas ou minúsculas.
4. 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.

EXERCÍCIOS

5. Dado uma string com uma frase informada pelo usuário (incluindo espaços em branco), conte:
 - a) quantos espaços em branco existem na frase.
 - b) quantas vezes aparecem as vogais a, e, i, o, u.
6. Faça um programa que leia um número de telefone, e corrija o número no caso deste conter somente 8 dígitos, acrescentando o '9' na frente. O usuário pode informar o número com ou sem o traço separador.
7. Desenvolva 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.