

Aprendendo ainda mais sobre Strings

Posicionamento de Strings

Aprendendo ainda mais sobre Strings

Posicionamento de Strings

Provavelmente você algum dia terá que apresentar o resultado do seu programa com strings alinhadas, tornando o resultado mais fácil de ler. Seja em um relatório ou simplesmente na saída do programa.

Aprendendo ainda mais sobre Strings


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 caracter informando como parâmetro para a função.

Aprendendo ainda mais sobre Strings

Posicionamento de Strings

```
>>> Texto1 = "Olá"
>>> Texto2 = "Mundo"
>>> Texto3 = Texto1.center(10)
>>> Texto4 = Texto2.center(10)
>>> Texto3
'  Olá  '
>>> Texto4
'  Mundo  '
>>> print(Texto3)
  Olá
>>> print(Texto4)
  Mundo
>>> print(Texto3 + "\n" + Texto4)
  Olá
  Mundo
>>> |
```

 python.exe

```
>>> Texto1 = "Olá"
>>> Texto2 = "Mundo"
>>> Texto3 = Texto1.center(10, '-')
>>> Texto4 = Texto2.center(10, '-')
>>> print(Texto3 + "\n" + Texto4)
---Olá---
--Mundo---
>>> |
```

Aprendendo ainda mais sobre Strings

Posicionamento de Strings

O Python também possui métodos para alinhar o texto à esquerda e à direita. São os métodos ***ljust*** e ***rjust***. 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.

Aprendendo ainda mais sobre Strings

Posicionamento de Strings

```
>>> Texto1 = "Olá"  
>>> Texto2 = "Mundo"  
>>> Texto3 = Texto1.ljust(5)  
>>> print(f"*{Texto3}*\n*{Texto2}*")  
*Olá  *  
*Mundo*  
>>>
```

Colocou 2 espaços à direita e alinhou o texto à esquerda.

```
>>> Texto1 = "Olá"  
>>> Texto2 = "Mundo"  
>>> Texto3 = Texto1.rjust(5)  
>>> print(f"*{Texto3}*\n*{Texto2}*")  
*  Olá*  
*Mundo*  
>>>
```

Colocou 2 espaços à esquerda e alinhou o texto à direita.

FIM