Manipulação de Strings

- **str.find(umaString, substring, inicio, fim)**: Retorna o índice da primeira ocorrência de substring.

  - *inicio* e *fim* são opcionais e indicam os intervalos de índices onde a busca será efetuada. Os *defaults* são, respectivamente, 0 e comprimento da string.
  - Caso substring não apareça na string, é retornado -1.
  - Note que o operador *in* pode ser usado para dizer se uma substring aparece numa string.

  Exemplo

```
>>> s = "quem parte e reparte, fica com a maior  parte"
>>> str.find(s, "parte")
5
>>> str.find(s, "reparte")
13
>>> str.find(s, "parcela")
-1
```

- **str.partition(umaString, sep)**: divide uma string em 3 partes : o que vem antes de *sep*, *sep* e o que vem após *sep*.

  Caso *sep* não seja encontrado, a string é retornada seguida por duas strings vazias.

  Exemplo

```
>>> s = "quem parte e reparte, fica com a maior  parte"

>>> str.partition(s,"t")
('quem par', 't', 'e e reparte, fica com a maior  parte')

>>>s.partition("z")
('quem parte e reparte, fica com a maior  parte', '', '')
```

- **str.join(umaString,sequencia)**: retorna uma string com todos os elementos da sequencia concatenados.

  Observação: Os elementos da sequência têm que ser strings.

  Importante: A string objeto é usada como separador entre os elementos.

  Exemplo

```
>>> str.join("/",("usr","bin","python"))   ou str.join("/",["usr","bin","python\])
'usr/bin/python'
>>> str.join("Q",(1,2,3,4,5))
TypeError: sequence item 0: expected string,
int found
>>> str.join("Q",('1','2','3','4','5'))
'1Q2Q3Q4Q5'
>>> str.join(Q,('1','2','3','4','5'))
```

```
Traceback (most recent call last):
  File "<pyshell#3>", line 1, in <module>
    Q.join((’1’,’2’,’3’,’4’,’5’))
NameError: name ’Q’ is not defined
```

- **str.replace(umaString,velho,novo,n)**: substitui as n instâncias da string *velho* por *novo*. Se n não for definido, todas as trocas são feitas.

  Exemplo

```
>>> s = "quem parte e reparte, fica com a maior parte"

>>> str.replace(s,"parte","parcela")
’quem parcela e reparcela, fica com a maior parcela’

>>> str.replace(s, "parte","parcela",2)
’quem parcela e reparcela, fica com a maior parte’
```

- **str.split(umaString, separador)**: retorna uma lista com as substrings presentes entre cópias da string separador.

  Se separador não for especificado, é assumido sequências de caracteres em branco, tabs ou newlines.

  Exemplo

```
>>> s = "xxx yyy zzz xxx yyy zzz"

>>> str.split(s)
  [’xxx’, ’yyy’, ’zzz’, ’xxx’, ’yyy’, ’zzz’]

>>> str.split(s, ’zzz’)
  [’xxx yyy ’, ’ xxx yyy ’, ’’]
```

- **str.strip(umaString, ch)**: retorna a string sem caracteres iniciais ou finais que estejam na string *ch*. Se *ch* não for especificada, retira caracteres em branco.

  Pode-se também usar *rstrip()* e *lstrip()* para retirar caracteres, respectivamente, à direita (final) ou à esquerda(início).

  Exemplo

```
>>> str.strip("   xxx afdsfa   ")
   ’xxx afdsfa’

>>> str.strip("xxx yyy zzz xxx","xy ")
   ’zzz’

>>> str.rstrip("   xxx")
   ’ xxx’
```