Python Bootcamp & Masterclass

string methods 3

gknxt

.partition( )

The `.partition(separator)` method splits the string at the first occurrence of `separator`, and returns a 3-tuple containing the part before the `separator`, the `separator` itself, and the part after the `separator`. If the `separator` is not found, it returns a 3-tuple containing the string itself, followed by two empty strings.

```
file_name = 'cust_data.json'
file_name.partition('.')
file_name.partition('/')
```

```
('cust_data', '.', 'json')

('cust_data.json', '', '')
```

```
ws = 'https://www.gknxt.com'
ws.partition('//')
```

```
('https:', '//', 'www.gknxt.com')
```

.rpartition( )

The **.rpartition(separator)** method splits the string at the last occurrence of **separator**, and returns a 3-tuple containing the part before the **separator**, the **separator** itself, and the part after the **separator**. If the **separator** is not found, it returns a 3-tuple containing the string itself, followed by two empty strings.

```
path = '/usr/local/bin/chrome'
path.rpartition('/')
```

```
('/usr/local/bin', '/', 'chrome')
```

```
ws = 'https://www.gknxt.com'
ws.rpartition('https://')
```

```
('', 'https://', 'www.gknxt.com')
```

```
file_name = 'cust_data.json'
file_name.rpartition('/')
```

```
('', '', 'cust_data.json')
```

gknxt

.replace( )

The `.replace(old, new [,count])` method returns a copy of the string with all occurrences of substring `old` replaced by `new`. If the optional argument `count` is given, only the first `count` occurrences are replaced.

```
s = 'It is the island of Istanbul, Turkey'
s.replace('is', 'was')
s.replace('is', 'was', 2)
s.replace(' is ', ' was ', 2)
s.replace('is', '')
```

```
'It was the wasland of Istanbul, Turkey'

'It was the wasland of Istanbul, Turkey'

'It was the island of Istanbul, Turkey'

'It  the land of Istanbul, Turkey'
```

.zfill()

The **.zfill(width)** method returns a copy of the string left filled with ASCII '0' digits to make a string of length **width**. A leading sign prefix ('+' / '-') is handled by inserting the padding after the sign character rather than before. The original string is returned if **width** is less than or equal to len(s) – the length of the string.

```
amt = '42107'
amt.zfill(8)
```

```
'00042107'
```

```
title = 'Sir'
title.zfill(6)
```

```
'000Sir'
```

```
ht = '+8889'
ht.zfill(10)
```

```
'+000008889'
```

```
amt = '42107'
amt.zfill(0)
amt.zfill(-1)
amt.zfill(3)
```

```
'42107'

'42107'

'42107'
```

```
temp = '-451'
temp.zfill(8)
```

```
'-0000451'
```

.ljust()

The `.ljust(width[,fillchar])` method returns the string left justified in a string of length `width`. Padding is done using the specified `fillchar` (default is an ASCII space).

The original string is returned if `width` is less than or equal to len(s) – the length of the string.

```
s = 'America'
s.ljust(10)
s.ljust(5)
s.ljust(10, '*')
```

```
'America   '

'America'

'America***'
```

```
s = 'Brazil'
s.ljust(0)
s.ljust(-1)
```

```
'Brazil'

'Brazil'
```

.rjust()

The `.rjust(width[,fillchar])` method returns the string right justified in a string of length `width`. Padding is done using the specified `fillchar` (default is an ASCII space).

The original string is returned if `width` is less than or equal to len(s) – the length of the string.

```
s = 'America'
s.rjust(10)
s.rjust(5)
s.rjust(10, '*')
```

```
'   America'

'America'

'***America'
```

```
s = 'Brazil'
s.rjust(0)
s.rjust(-1)
```

```
'Brazil'

'Brazil'
```

.center()

The `.center(width[,fillchar])` method returns centered in a string of length `width`.

Padding is done using the specified `fillchar` (default is an ASCII space).

The original string is returned if width is less than or equal to len(s) – the length of the string.

```
s = 'America'
s.center(10)
s.center(10, '*')
s.center(11, '*')
```

```
' America  '

'*America**'

'**America**'
```

```
t = 'America'
t.center(5)
t.center(0)
t.center(-1)
```

```
'America'

'America'

'America'
```

.expandtabs()

The `.expandtabs(tabsize=8)` method returns a copy of the string where all tab characters are replaced by one or more spaces, depending on the current column and the given tab size. Tab positions occur every `tabsize` characters (default is 8, giving tab positions at columns 0, 8, 16 and so on)

```
s = '01\t012\t0123\t01234'
print(s)
s.expandtabs()
s.expandtabs(8)
s.expandtabs(4)
```

```
01      012     0123    01234

'01      012     0123    01234'

'01      012     0123    01234'

'01  012 0123    01234'
```

```
u =  'USA\t04\tJuly\t1776\r\n'
u.expandtabs(4)
u.expandtabs(8)
```

```
'USA 04  July    1776\r\n'

'USA     04      July    1776\r\n'
```

.removeprefix()

The `.removeprefix(prefix)` method returns string after removing the `prefix`, if the string starts with the `prefix` and that `prefix` is not empty Otherwise, returns a copy of the original string itself.

```
s = 'Author: Alexander Hamilton'
s.removeprefix('Author: ')
s.removeprefix('author:')
s.lstrip('Author: ')
```

```
'Alexander Hamilton'

'Author: Alexander Hamilton'

'lexander Hamilton'
```

.removesuffix()

The `.removesuffix(suffix)` method returns string after removing the `suffix`, if the string ends with the `suffix` and that `suffix` is not empty. Otherwise, returns a copy of the original string itself.

```
t = 'Microsoft test'
t.removesuffix('test')
t.removesuffix(' test')
t.removesuffix('Test')
```

```
'Microsoft '

'Microsoft'

'Microsoft test'
```

Online Resources

For best python resources, please visit:

gknxt.com/python/

gknxt

Python Bootcamp & Masterclass

Thank You

for your Rating & Review

gknxt