TOOL

String Manipulation Methods

Case Conversion Methods

Method	Definition	Example	Output
`str.lower()`	Returns the input string with all cased characters converted to lowercase.	`'I like NLP'.lower()`	`'i like nlp'`
`str.upper()`	Returns the input string with all cased characters converted to uppercase.	`'I like NLP'.upper()`	`'I LIKE NLP'`
`str.capitalize()`	Returns the string with its first character capitalized and the rest lowercased.	`'we like NLP'.capitalize()`	`'We like nlp'`
`str.title()`	Returns the string where each word starts with an uppercase character and the remaining characters are lowercase.	`'we like NLP'.title()`	`'We Like Nlp'`

Replacement Methods

Method	Definition	Example	Output
`str.replace(old, new [,count])`	Returns the string with all occurrences of substring **`old`** replaced by **`new`**. The optional argument **`count`** only replaces the first count occurrences. to lowercase.	`'I like NLP'. replace('I', 'we')`	`'we like NLP'`

Test Methods

Method	Definition	Example	Output
`str.isalpha()`	Returns **`True`** if the string contains at least one character and all characters in the string are alphabetic. If not, it returns **`False`**.	`'nlp'.isalpha()` `'nlp1'.isalpha()`	`True` `False`

`str.isalnum()`	Returns **`True`** if the string contains at least one character and all characters in the string are alphanumeric, and otherwise returns **`False`**.	`'nlp'.isalnum()`
		`'nlp123'.isalnum()`
		`'1+3'.isalnum()`
			`True`
			`True`
			`False`

`str.isdecimal()`	Returns **`True`** if the string contains at least one decimal character and all characters in the string are decimal characters, and returns **`False`** otherwise.	`'12345'.isdecimal()` `'nlp123'.isdecimal()`	`True` `False`

Split/Join/Strip Methods

Method	Definition	Example	Output
`str.split (sep=None, maxsplit=-1)`	Returns a list of the words in the string, using **sep** as the delimiter string. If **maxsplit** is given a numeric value, then at most that number of splits are done (thus, the list will have at most **maxsplit+1** elements). If **maxsplit** is not specified or set to **-1**, then all possible splits are made.	`'I,like,NLP' .split(',')`	`['I', 'like' , 'NLP']`
`str.join(iterable)`	Returns a string which is the concatenation of the strings in **iterable**. The separator between elements is the string providing this method.	`' '.join('I like NLP'.split(' '))`	`'I like NLP'`
`str.strip([chars])`	Returns a copy of the string with the leading and trailing characters removed. The optional **chars** argument is a string specifying the set of characters to be removed. If omitted or set to **None**, the default is used, which removes whitespace.	`'  NLP  '.strip()`	`'NLP'`