String Advanced Methods

isnumeric(): This method returns True if all the characters in a string are numeric.

In [ ]:
```
string = "12345"
print(string.isnumeric()) # output: True

string = "12ab3"
print(string.isnumeric()) # output: False
```

```
True
False
```

isprintable(): This method returns True if all the characters in a string are printable.

In [ ]:
```
string = "Hello, World!\n"
print(string.isprintable()) # output: False

string = "Hello, World!"
print(string.isprintable()) # output: True
```

```
False
True
```

isspace(): This method returns True if all the characters in a string are whitespace.

In [ ]:
```
string = "   "
print(string.isspace()) # output: True

string = "  Hello  "
print(string.isspace()) # output: False
```

```
True
False
```

istitle(): This method returns True if the string follows the rules of title case.

In [ ]:
```
string = "Hello, World"
print(string.istitle()) # output: True

string = "hello, world"
print(string.istitle()) # output: False
```

```
True
False
```

isupper(): This method returns True if all the characters in a string are uppercase.

In [ ]:
```
string = "HELLO, WORLD"
print(string.isupper()) # output: True

string = "Hello, World"
print(string.isupper()) # output: False
```

```
True
False
```

join(): This method joins a sequence of strings into a single string.

In [ ]:
```
list_of_strings = ["hello", "world"]
delimiter = " "
string = delimiter.join(list_of_strings)
print(string) # output: "hello world"
```

```
hello world
```

ljust(): This method pads a string with a specified character on the left until it reaches a specified width.

In [ ]:
```
string = "hello"
width = 10
padding_char = "-"
padded_string = string.ljust(width, padding_char)
print(padded_string) # output: "hello-----"
```

```
hello-----
```

lower(): This method returns a string in lowercase.

In [ ]:
```
string = "HELLO"
print(string.lower()) # output: "hello"
```

```
hello
```

maketrans(): This method returns a translation table that can be used with the translate() method to replace specified characters.

In [ ]:
```
old_chars = "aeiou"
new_chars = "12345"
translation_table = str.maketrans(old_chars, new_chars)
string = "hello world"
print(string.translate(translation_table)) # output: "h2ll4 w4rld"
```

```
h2ll4 w4rld
```

partition(): This method splits a string into a tuple containing the part before the first occurrence of a specified substring, the substring itself, and the part after.

In [ ]:
```
string = "hello world"
print(string.partition(" ")) # output: ("hello", " ", "world")
```

```
('hello', ' ', 'world')
```

removeprefix(): This method removes a specified prefix from a string.

In [ ]:
```
string = "hello world"
prefix = "hello"
print(string.removeprefix(prefix)) # output: " world"
```

```
 world
```

removesuffix(): This method removes a specified suffix from a string.

In [ ]:
```
string = "hello world"
suffix = "world"
```

```
print(string.removesuffix(suffix)) # output: "hello "
```

```
hello
```

replace(): This method replaces all occurrences of a specified substring with another substring.

In [ ]:
```
string = "hello world"
old_substring = "world"
new_substring = "python"
print(string.replace(old_substring, new_substring)) # output: "hello python"
```

```
hello python
```

rfind(): This method returns the index of the last occurrence of a specified substring in a string.

In [ ]:
```
string = "hello world"
substring = "o"
print(string.rfind(substring)) # output: 7
```

```
7
```

rindex(): This method returns the index of the last occurrence of a specified substring in a string, or raises a ValueError if the substring is not found.

In [ ]:
```
string = "hello world"
substring = "o"
print(string.rindex(substring)) # output: 7

string = "hello world"
substring = "x"
print(string.rindex(substring)) # ERROR: raises ValueError
```

```
7
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
j:\01_Repos\Python_Full\01_Python\02_Fundamentals\02_3_String_methods.ipynb Cel
l 31 in 7
      <a href='vscode-notebook-cell:/j%3A/01_Repos/Python_Full/01_Python/02_Fun
damentals/02_3_String_methods.ipynb#X42sZmlsZQ%3D%3D?line=4'>5</a> string = "he
llo world"
      <a href='vscode-notebook-cell:/j%3A/01_Repos/Python_Full/01_Python/02_Fun
damentals/02_3_String_methods.ipynb#X42sZmlsZQ%3D%3D?line=5'>6</a> substring =
"x"
----> <a href='vscode-notebook-cell:/j%3A/01_Repos/Python_Full/01_Python/02_Fun
damentals/02_3_String_methods.ipynb#X42sZmlsZQ%3D%3D?line=6'>7</a> print(strin
g.rindex(substring))

ValueError: substring not found
```

rjust(): This method pads a string with a specified character on the right until it reaches a specified width.

In [ ]:
```
string = "hello"
width = 10
padding_char = "-"
```

```
padded_string = string.rjust(width, padding_char)
print(padded_string) # output: "-----hello"
```

```
-----hello
```

rpartition(): This method splits a string into a tuple containing the part before the last occurrence of a specified substring, the substring itself, and the part after.

In [ ]:
```
string = "hello world"
print(string.rpartition(" ")) # output: ("hello", " ", "world")
```

```
('hello', ' ', 'world')
```

rsplit(): This method splits a string into a list of substrings, starting from the right.

In [ ]:
```
string = "hello world"
print(string.rsplit(" ")) # output: ["hello", "world"]
```

```
['hello', 'world']
```

rstrip(): This method removes any whitespace characters from the end of a string.

In [ ]:
```
string = "   hello   "
print(string.rstrip()) # output: "   hello"
```

```
   hello
```

split(): This method splits a string into a list of substrings.

In [ ]:
```
string = "hello world"
print(string.split(" ")) # output: ["hello", "world"]
```

```
['hello', 'world']
```

splitlines(): This method splits a string into a list of lines.

In [ ]:
```
string = "hello\nworld"
print(string.splitlines()) # output: ["hello", "world"]
```

```
['hello', 'world']
```

startswith(): This method returns True if a string starts with a specified substring.

In [ ]:
```
string = "hello world"
substring = "hello"
print(string.startswith(substring)) # output: True
```

```
True
```

strip(): This method removes any whitespace characters from the beginning and end of a string.

In [ ]:
```
string = "   hello   "
print(string.strip()) # output: "hello"
```

```
hello
```

swapcase(): This method swaps the case of all the characters in a string.

In [ ]:
```
string = "Hello, World"
print(string.swapcase()) # output: "hELLO, wORLD"
```

```
hELLO, wORLD
```

title(): This method returns a string in title case.

In [ ]:
```
string = "hello, world"
print(string.title()) # output: "Hello, World"
```

```
Hello, World
```

translate(): This method replaces specified characters in a string using a translation table.

In [ ]:
```
old_chars = "aeiou"
new_chars = "12345"
translation_table = str.maketrans(old_chars, new_chars)
string = "hello world"
print(string.translate(translation_table)) # output: "h2ll4 w4rld"
```

```
h2ll4 w4rld
```