Manipulating Strings

Escape Characters

In [5]:

```
print("Hello there!\nHow are you?\nI\'m doing fine.")
```

```
Hello there!
How are you?
I'm doing fine.
```

In [7]:

```
print(r"Hello there!\nHow are you?\nI\'m doing fine.")
```

```
Hello there!\nHow are you?\nI\'m doing fine.
```

Multiline Strings

In [13]:

```
print
("""Dear Alice,

Eve's cat has been arrested for catnapping,
cat burglary, and extortion.

Sincerely,
Bob"""

)
```

Out[13]:

```
"Dear Alice, \n\nEve's cat has been arrested for catnapping,\ncat burglary, and
extortion.\n\nSincerely,\nBob"
```

Indexing and Slicing strings

Indexing

In [17]:

```
spam ='Hello world!'

spam[0]
```

Out[17]:

```
'H'
```

In [19]:

```
spam[4]
```

Out[19]:

```
'o'
```

In [21]:

```
spam[-1]
```

Out[21]:

```
'!'
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

Slicing

```
In [24]: spam='Hello world!'
```

```
In [26]: spam[0:5]
```

```
Out[26]: 'Hello'
```

```
In [32]: spam[6:12]
```

```
Out[32]: 'world!'
```

```
In [34]: spam[6:-1]
```

```
Out[34]: 'world'
```

```
In [36]: spam[:-1]
```

```
Out[36]: 'Hello world'
```

```
In [38]: spam[::-1]
```

```
Out[38]: '!dlrow olleH'
```

```
In [40]: fizz=spam[0:5]
```

```
In [42]: fizz
```

```
Out[42]: 'Hello'
```

The in and not in operators

```
In [45]: 'Hello' in 'Hello World'
```

```
Out[45]: True
```

```
In [47]: 'HELLO' in 'Hello World'
```

```
Out[47]: False
```

```
In [49]: '' in 'spam'
```

```
Out[49]: True
```

```
In [51]: 'cats' not in 'cats and dogs'
```

```
Out[51]: False
```

upper(), lower() and title()

```
In [54]: greet='Hello world!'
         greet.upper()
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

```
Out[54]:  'HELLO WORLD!'
```

```
In [56]:  greet.lower()
```

```
Out[56]:  'hello world!'
```

```
In [60]:  greet.title()
```

```
Out[60]:  'Hello World!'
```

isupper() and islower() methods

```
In [63]:  spam='Hello world!'
```

```
In [65]:  spam.islower()
```

```
Out[65]:  False
```

```
In [67]:  spam.isupper()
```

```
Out[67]:  False
```

```
In [69]:  'HELLO!'.isupper()
```

```
Out[69]:  True
```

```
In [71]:  'abc12345'.islower()
```

```
Out[71]:  True
```

```
In [73]:  '12345'.islower()
```

```
Out[73]:  False
```

```
In [75]:  '12345'.isupper()
```

```
Out[75]:  False
```

The isX string methods

isalpha(), isalnum(), isdecimal(), isspace(), istitle()

```
In [90]:  string="Hello World"
```

```
In [92]:  string.isalpha()
```

```
Out[92]:  False
```

```
In [94]:  string.isspace()
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

```
In [96]:  string.istitle()

Out[96]:  True

In [114…  num="Sider1234.356man"

In [116…  num.isdecimal()

Out[116…  False

In [118…  num.isalnum()

Out[118…  False

In [120…  string

Out[120…  'Hello World'

In [122…  string.isalnum()

Out[122…  False
```

startswith() and endswith()

```
In [125…  'Hello world!'.startswith('Hello')

Out[125…  True

In [129…  'Hello world!'.endswith('world!')

Out[129…  True

In [133…  'Hello world!'.endswith('World!')

Out[133…  False
```

join() and split()

join()

```
In [147…  ''.join(['My','name','is','Sid'])

Out[147…  'MynameisSid'

In [145…  ', '.join(['cats','rats','bats'])

Out[145…  'cats, rats, bats'

In [149…  ' '.join(['My','name','is','Sid'])
```

Out[149…
```
'My name is Sid'
```

In [151…
```python
'123'.join(['My','name','is','Sid'])
```

Out[151…
```
'My123name123is123Sid'
```

In [153…
```python
"My name is Sid".split()
```

Out[153…
```
['My', 'name', 'is', 'Sid']
```

In [157…
```python
'MyABCnameABCisABCSid'.split('ABC')
```

Out[157…
```
['My', 'name', 'is', 'Sid']
```

In [161…
```python
"My name is Sid".split('m')
```

Out[161…
```
['My na', 'e is Sid']
```

In [167…
```python
" My  name  is  Sid ".split(' ')
```

Out[167…
```
['', 'My', '', 'name', '', 'is', '', 'Sid', '']
```

Justifying text with rjust(), ljust() and center()

In [170…
```python
'Hello'.rjust(10)
```

Out[170…
```
'     Hello'
```

In [172…
```python
'Hello'.rjust(20)
```

Out[172…
```
'               Hello'
```

In [174…
```python
'Hello'.ljust(10)
```

Out[174…
```
'Hello     '
```

In [176…
```python
'Hello'.center(20)
```

Out[176…
```
'       Hello        '
```

In [178…
```python
'Hello'.rjust(20,'*')
```

Out[178…
```
'***************Hello'
```

In [180…
```python
'Hello'.ljust(20,'-')
```

Out[180…
```
'Hello---------------'
```

In [184…
```python
'Hello'.center(20,'$')
```

Out[184…
```
'$$$$$$$Hello$$$$$$$$'
```

Removing whitespace with strip(), rstrip(), and lstrip()

In [191…
```
spam="  Hello    world  "
```

In [193…
```
spam.strip()
```

Out[193…
```
'Hello    world'
```

In [195…
```
spam.lstrip()
```

Out[195…
```
'Hello    world  '
```

In [197…
```
spam.rstrip()
```

Out[197…
```
'  Hello    world'
```

In [209…
```
spam='SpamSpamBaconSpamEggsSpamSpam'
spam.strip('apmS')
```

Out[209…
```
'BaconSpamEggs'
```

The Count Method

In [212…
```
sentence='one sheep two sheep three sheep four'
sentence.count('sheep')
```

Out[212…
```
3
```

In [214…
```
sentence.count('e')
```

Out[214…
```
9
```

In [218…
```
sentence.count('e',6) # returns count of e after 'one sh'
```

Out[218…
```
8
```

In [220…
```
sentence.count('e',7)
```

Out[220…
```
7
```

In [222…
```
sentence.count('e',9)
```

Out[222…
```
6
```

Replace Method

In [225…
```
text="Hello, world!"
                                            )
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

Out[225… 'Hello, planet!'

In [237…
```
fruits = "apple, banana, cherry, apple"
fruits.replace("apple", "orange",1)
```

Out[237… 'orange, banana, cherry, apple'

In [229…
```
fruits = "apple, banana, cherry, apple"
fruits.replace("apple", "orange", 2)
```

Out[229… 'orange, banana, cherry, orange'

In [239…
```
fruits = "apple, banana, cherry, apple"
fruits.replace("apple", "orange")
```

Out[239… 'orange, banana, cherry, orange'

In [233…
```
sentence = "I like apples, Apples are my favorite fruit"
sentence.replace("apples", "oranges")
```

Out[233… 'I like oranges, Apples are my favorite fruit'

Strings more operations

Cases

In [243…
```
s='hello'
s.capitalize()
```

Out[243… 'Hello'

In [245…
```
s='HELLO'
s.lower()
```

Out[245… 'hello'

In [249…
```
s='Hello'
s.swapcase()
```

Out[249… 'hELLO'

In [255…
```
s='hello world'
s.title()
```

Out[255… 'Hello World'

In [257…
```
s='hello'
s.upper()
```

Out[257… 'HELLO'

Sequence Operations

In [262…

```
s='Hello'
s2='el'
s2 in s
```

Out[262…

```
True
```

In [264…

```
s='Hello'
s2='World'
s+s2
```

Out[264…

```
'HelloWorld'
```

In [266…

```
len(s)
```

Out[266…

```
5
```

In [268…

```
min(s)
```

Out[268…

```
'H'
```

In [270…

```
max(s)
```

Out[270…

```
'o'
```

In [272…

```
s2 not in s
```

Out[272…

```
True
```

In [274…

```
s*3
```

Out[274…

```
'HelloHelloHello'
```

In [276…

```
s[-1]
```

Out[276…

```
'o'
```

In [278…

```
s[1:4:2]
```

Out[278…

```
'el'
```

In [286…

```
s.count(s2)
```

Out[286…

```
0
```

In [294…

```
s3='l'
```

In [296…

```
s.count(s3)
```

Out[296…

```
2
```

In [298…

```
s4='Hell'
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

Out[298…

```
1
```

In [300…

```
s5='ell'
s.count(s5)
```

Out[300…

```
1
```

In [302…

```
s.count(s2)
```

Out[302…

```
0
```

Whitespace

In [305…

```
s='hi'
s.center(20)
```

Out[305…

```
'         hi         '
```

In [307…

```
s.isspace()
```

Out[307…

```
False
```

In [309…

```
s.ljust(20)
```

Out[309…

```
'hi                  '
```

In [311…

```
s.rjust(20)
```

Out[311…

```
'                  hi'
```

In [313…

```
s="      Hii My name is Sid    "
s.strip()
```

Out[313…

```
'Hii My name is Sid'
```

Find / Replace

In [327…

```
s='Hello Othello'
s2='o'
s.index(s2,0,5)
```

Out[327…

```
4
```

In [329…

```
s.find(s2)
```

Out[329…

```
4
```

In [331…

```
s.index(s2)
```

Out[331…

```
4
```

In [333…

```
s3='P'
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

```
Out[333…  'HellP OthellP'
```

```
In [335…  s.replace(s2,s3,1)
```

```
Out[335…  'HellP Othello'
```

```
In [337…  s.replace(s2,s3,5)
```

```
Out[337…  'HellP OthellP'
```

```
In [339…  s.rfind(s2)
```

```
Out[339…  12
```

```
In [341…  s4='X'
          s.rfind(s4)
```

```
Out[341…  -1
```

```
In [343…  s.rindex(s4)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[343], line 1
----> 1 s.rindex(s4)

ValueError: substring not found
```

Cases-2

```
In [347…  s
```

```
Out[347…  'Hello Othello'
```

```
In [349…  s.casefold()
```

```
Out[349…  'hello othello'
```

```
In [351…  s='123hr'
          s.casefold()
```

```
Out[351…  '123hr'
```

Splitting

```
In [354…  s="My name is Sid"
```

```
In [356…  s.join('123')
```

```
Out[356…  '1My name is Sid2My name is Sid3'
```

```
In [360…  '123'.join(['My', 'Name', 'is', 'Sid'])
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

Out[360…

```
'My123Name123is123Sid'
```

In [362…

```
s.partition(s)
```

Out[362…

```
('', 'My name is Sid', '')
```

In [372…

```
s.partition(' ')
```

Out[372…

```
('My', ' ', 'name is Sid')
```

In [376…

```
s.rpartition(' ')
```

Out[376…

```
('My name is', ' ', 'Sid')
```

In [386…

```
s.rsplit(' ', 4)
```

Out[386…

```
['My', 'name', 'is', 'Sid']
```

In [388…

```
s.rsplit(' ', 2)
```

Out[388…

```
['My name', 'is', 'Sid']
```

In [390…

```
s.rsplit(' ', 1)
```

Out[390…

```
['My name is', 'Sid']
```

In [392…

```
s.split(' ', 1)
```

Out[392…

```
['My', 'name is Sid']
```

In [394…

```
s.rsplit(' ', 3)
```

Out[394…

```
['My', 'name', 'is', 'Sid']
```

In [396…

```
s.splitlines()
```

Out[396…

```
['My name is Sid']
```

In [402…

```
s=""" My

name

is



Sid Bose

"""
s.splitlines()
```

Out[402…

```
[' My', '', 'name', '', 'is ', '', '', '', 'Sid Bose', '']
```

Inspection II

In [405…

```
s
s[2:5]
```

Out[405…

```
'y\n\n'
```

In [417…

```
s.endswith(('ose',"bose","\n"))
```

Out[417…

```
True
```

In [419…

```
s
```

Out[419…

```
' My\n\nname\n\nis \n\n\n\nSid Bose\n\n'
```

In [421…

```
s.isidentifier()
```

Out[421…

```
False
```

In [423…

```
s="Myname"
s.isidentifier()
```

Out[423…

```
True
```

In [425…

```
s="Myname123"
s.isidentifier()
```

Out[425…

```
True
```

In [427…

```
s.isprintable()
```

Out[427…

```
True
```

Whitespace II

In [430…

```
s
```

Out[430…

```
'Myname123'
```

In [444…

```
s.center(15,'$')
```

Out[444…

```
'$$$Myname123$$$'
```

In [446…

```
s.expandtabs(10)
```

Out[446…

```
'Myname123'
```

In [454…

```
s.zfill(15)
```

Out[454…

```
'000000Myname123'
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js