Manipulating Strings

Escape character

In [86]:
```
>>> print("Hello John!\nHow are you?\nI\'m doing fine.")
```

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

Raw strings

A raw string entirely ignores all escape characters and prints any backslash that appears in the string

In [90]:
```
>>> print(r"Hello Peter!\nHow are you?\nI\'m doing fine.")
```

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

In [ ]:
```

```

Raw strings are mostly used for regular expression defination.

Multiline Strings

In [103…
```
>>> print(
... """Dear Alice,
...
... Eve's cat has been arrested for catnapping,
... cat burglary, and extortion.
...
... Sincerely,
... Rohit"""
... )
```

```
Dear Alice,

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

Sincerely,
Rohit
```

In [ ]:
```

```

Indexing and Slicing strings

H e l l o w o r l d !

0 1 2 3 4 5 6 7 8 9 10 11

Indexing

In [119…
```
>>> spam = 'Hello world!'
```

In [133…
```
>>> spam[0]
```

Out[133… 'h'

In [135…
```
>>> spam[4]
```

Out[135… 'o'

In [125…
```
spam[-1]
```

Out[125… '!'

In [137…
```
>>> spam[-5]
```

Out[137… 'o'

Slicing

In [141…
```
>>> spam = 'Hello world!'
```

In [143…
```
>>> spam[0:5]
```

Out[143… 'Hello'

In [145…
```
>>> spam[1:5]
```

Out[145… 'ello'

In [147…
```
>>> spam[6:-1]
```

Out[147… 'world'

In [151…
```
>>> spam[:-1]
```

Out[151… 'Hello world'

In [153…
```
>>> spam[::-1]
```

Out[153… '!dlrow olleH'

In [155…
```
>>> fizz = spam[0:5]
>>> fizz
```

Out[155… 'Hello'

The in and not in operators

In [158…
```
>>> 'Hello' in 'Hello world'
```

Out[158… True

In [160…
```
>>> 'Hello' in 'Hello'
```

Out[160… True

In [162…
```
>>> '' in 'spam'
```

Out[162… True

In [164…
```
>>> 'cats' not in 'cats and dogs'
```

Out[164… False

upper(),lower() and title()

Transforms a string to upper, lower and title case:

In [176…
```
>>> greet = 'Hello world!'
>>> greet.upper()
```

Out[176… 'HELLO WORLD!'

In [178…
```
>>> greet.lower()
```

Out[178… 'hello world!'

In [180…
```
>>> greet.title()
```

Out[180… 'Hello World!'

In [ ]:

isupper(), islower() methods

Returns 'True' or 'False' after evaluating if a string is in upper or lower case:

In [184…
```
>>> spam = 'Hello world!'
>>> spam.islower()
```

Out[184… False

In [186…
```
>>> spam.isupper()
```

Out[186… False

In [192…
```
>>> 'HELLO'.isupper()
```

Out[192… True

In [194…
```
>>> 'abc12345'.islower()
```

Out[194… True

In [196…
```
>>> '12345'.islower()
```
Out[196…
```
False
```

In [198…
```
>>> '12345'.isupper()
```
Out[198…
```
False
```

The isX string methods

Method Description

isalpha() returns `True` if the string consists only of letters.

isalnum() returns `True` if the string consists only of letters and numbers.

isdecimal() returns `True` if the string consists of numbers.

isspace() retuns `True` if the string consists only of spaces, tabs and new-lines.

istitle() returns `True` if the string consists only of words that begin with an uppercase letter followed by only lowercase characters.

startswith() and endswith()

In [214…
```
>>> 'Hello World!'.startswith('Hello')
```
Out[214…
```
True
```

In [216…
```
>>> 'Hello World!'.endswith('World!')
```
Out[216…
```
True
```

In [218…
```
>>> 'abc123'.startswith('abc')
```
Out[218…
```
True
```

In [220…
```
>>> 'abc123'.startswith('abcdef')
```
Out[220…
```
False
```

In [222…
```
>>> 'abc123'.endswith('12')
```
Out[222…
```
False
```

In [224…
```
>>> 'Hello world!'.startswith('Hello world!')
```
Out[224…
```
True
```

In [230…
```
>>> 'Hello world!'.endswith('Hello world!')
```
Out[230…
```
True
```

join() and split()

join()

The `join()` method takes all the items in an iterable, like a list, dictionary, tuple or set and joins them into a string. You can also specify a separator.

In [242…
```
>>> ''.join(['My', 'name', 'is', 'Devil'])
```
Out[242…
```
'MynameisDevil'
```

In [240…
```
>>> ', '.join(['cats', 'dogs', 'bats'])
```
Out[240…
```
'cats, dogs, bats'
```

In [244…
```
>>> ' '.join(['My', 'name', 'is', 'Devil'])
```
Out[244…
```
'My name is Devil'
```

In [246…
```
'ABC'.join(['My', 'name', 'is', 'Devil'])
```
Out[246…
```
'MyABCnameABCisABCDevil'
```

Split()

The `split()` `method splits a` string `into a` list`. By default, it will use whitespace to separate the items, but you can also set another character of choice:

In [261…
```
>>> 'My name is Rohit'.split()
```
Out[261…
```
['My', 'name', 'is', 'Rohit']
```

In [263…
```
>>> 'MyABCnameABCisABCRohit'.split('ABC')
```
Out[263…
```
['My', 'name', 'is', 'Rohit']
```

In [267…
```
>>> 'My name is Simon'.split('m')
```
Out[267…
```
['My na', 'e is Si', 'on']
```

In [271…
```
>>> 'My name is Simon'.split()
```
Out[271…
```
['My', 'name', 'is', 'Simon']
```

In [279…
```
>>> 'My name is Rohit'.split(' ')
```
Out[279…
```
['My', 'name', 'is', 'Rohit']
```

Justifying text with rjust(), ljust() andcenter()

In [285…
```
>>> 'Hello'.rjust(10)
```
Out[285…
```
'     Hello'
```
In [287…
```
>>> 'Hello'.rjust(20)
```
Out[287…
```
'               Hello'
```
In [289…
```
>>> 'Hello World'.rjust(20)
```
Out[289…
```
'         Hello World'
```
In [293…
```
>>> 'Hello World'.ljust(20)
```
Out[293…
```
'Hello World         '
```
In [295…
```
>>> 'Hello'.ljust(10)
```
Out[295…
```
'Hello     '
```
In [298…
```
>>> 'Hello'.center(10)
```
Out[298…
```
'  Hello   '
```

An optional second argument to `rjust()` and `ljust()` will specify a fill character apart from a space character:

In [301…
```
>>> 'Hello'.rjust(20,'*')
```
Out[301…
```
'***************Hello'
```
In [303…
```
>>> 'Hello'.ljust(20,'-')
```
Out[303…
```
'Hello---------------'
```
In [309…
```
>>> 'Hello'.center(20, '=')
```
Out[309…
```
'=======Hello========'
```

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

In [312…
```
>>> spam = '    Hello World    '
>>> spam.strip()
```
Out[312…
```
'Hello World'
```
In [314…
```
>>> spam.lstrip()
```
Out[314…
```
'Hello World    '
```
In [316…
```
>>> spam.rstrip()
```

Out[316… '   Hello World'

In [320…
```
>>> spam = 'SpamSpamBaconSpamEggsSpamSpam'
>>> spam.strip('ampS')
```

Out[320… 'BaconSpamEggs'

The Count Method

Counts the number of occurences of a given character or substring in the string it is applied to can be optionally provided start and end index.

In [324…
```
>>> sentence = 'one lion two lion three lion four'
>>> sentence.count('lion')
```

Out[324… 3

In [328…
```
>>> sentence.count('o')
```

Out[328… 6

In [334…
```
>>> sentence.count('l', 4)
```

Out[334… 3

Replace Method

Replaces all occurences of a given substring with anotheer substring. Can be optionally provided a third argument to limit the number of replacements. Returns a new string

In [338…
```
>>> text = "Hello World!"
>>> text.replace("World", "Planet")
```

Out[338… 'Hello Planet!'

In [346…
```
>>> fruits = "apple, banana, mango, orange, apple"
>>> fruits.replace("apple", "strawberry", 1)
```

Out[346… 'strawberry, banana, mango, orange, apple'

In [350…
```
>>> sentence = "I like apples, Mangoes are my favourite fruit"
>>> sentence.replace("apples", "pineapple")
```

Out[350… 'I like pineapple, Mangoes are my favourite fruit'

In [ ]: