String Functions

In [1]:
```
#Capitalize()
#converts first character to uppercase and others to lowercase
#capitalized_string = sentence.capitalize()
```

In [6]:
```
sen= 'hey chandan'
```

In [8]:
```
sen=sen.capitalize()
sen
```

Out[8]:
```
'Hey chandan'
```

In [13]:
```
sen1='hey amar'
```

In [14]:
```
sen1= sen1.capitalize()
sen1
```

Out[14]:
```
'Hey amar'
```

In [15]:
```
sen2= 'i love python'
```

In [16]:
```
sen2=sen2.capitalize()
sen2
```

Out[16]:
```
'I love python'
```

In [1]:
```
# Center()
# returns the centered padded string of length
# str.center(width, [fillchar])
```

In [2]:
```
x= 'How are you feeling'
```

In [5]:
```
x= x.center(24,'*')
x
```

Out[5]:
```
'**How are you feeling***'
```

In [6]:
```
y= 'Good morning guys'
y=y.center(20,'#')
y
```

Out[6]:
```
'#Good morning guys##'
```

In [7]:
```
# Casefold()
# convert all characters to lowercase
# str.casefold()
```

In [10]:
```
str= 'NO WORRIES'
```

In [11]:
```
str= str.casefold()
str
```

Out[11]:
```
'no worries'
```

In [12]:
```
str1= 'HEY THERE!'
```

In [13]:
```
str1= str1.casefold()
str1
```

Out[13]:
```
'hey there!'
```

In [14]:
```
# String count
# The count() method returns the number of occurrences of a substring in the given str
# string.count(substring, start=..., end=...)
```

In [18]:
```
a= 'python is popular nowadays'
```

In [20]:
```
count= a.count('p')
print("The count of 'P' is: ",count)
```

```
The count of 'P' is:  3
```

In [22]:
```
b= 'cuttack town is coming under cuttack district'
```

In [24]:
```
count_1= b.count('cuttack')
print("Here the number of cuttack is: ", count_1)
```

```
Here the number of cuttack is:  2
```

In [25]:
```
# Endswith()
# The endswith() method returns True if a string ends with the specified suffix. If no
# str.endswith(suffix[, start[, end]])
```

In [26]:
```
message= 'python is fun'
```

In [28]:
```
message.endswith('fun')
```

Out[28]:
```
True
```

In [30]:
```
message_1= 'I am from odisha'
```

In [32]:
```
message_1.endswith('sha')
```

Out[32]:
```
True
```

In [33]:
```
# expandtabs()
# The expandtabs() takes an integer tabsize argument. The default tabsize is 8.
# The expandtabs() returns a string where all '\t' characters are replaced with whites
# string.expandtabs(tabsize)
```

In [36]:
```
string= 'abc\t123\txyz'
```

In [37]:
```
result= string.expandtabs()
result
```

Out[37]:
```
'abc     123     xyz'
```

In [38]:
```
string_1= 'wish\tyou\ta\tvery\thappy\tbirthday'
```

In [39]:
```
result_1=string_1.expandtabs()
result_1
```

Out[39]:
```
'wish    you     a       very    happy   birthday'
```

In [40]:
```
# encode()
# The encode() method returns an encoded version of the given string.
# string.encode(encoding='UTF-8',errors='strict')
```

In [43]:
```
sen = 'Lōndōn diāries'
```

In [44]:
```
sen_utf= sen.encode()
```

In [45]:
```
print("The encoded version is: ",sen_utf)
```

```
The encoded version is:  b'L\xc5\x8dnd\xc5\x8dn di\xc4\x81ries'
```

In [51]:
```
sen_1= 'pythōn'
```

In [53]:
```
sen_1_utf= sen_1.encode()
```

In [54]:
```
print("The encoded version is: ",sen_1_utf)
```

```
The encoded version is:  b'pyth\xc5\x8dn'
```

In [55]:
```
# Find()
# The find() method returns the index of first occurrence of the substring (if found).
```

In [56]:
```
quote= 'let bygone be bygone'
```

In [60]:
```
result_1= quote.find('be')
result_1
```

Out[60]:
```
11
```

In [61]:
```
result_2=quote.find('sheep')
result_2
```

Out[61]:
```
-1
```

In [62]:
```
quote_1='little things matters a lot'
```

In [63]:
```
result_3= quote_1.find('a')
result_3
```

Out[63]:
```
15
```

In [64]:
```
# format()
# The format() method returns the formatted string.
```

In [65]:
```
# Default arguements
print("Hello {},your current balance is {}".format('Prashant',30000))
```

```
Hello Prashant,your current balance is 30000
```

In [66]:
```
print("Hey, {} ,can I call you {}".format('Subhendu','Subh'))
```

```
Hey, Subhendu ,can I call you Subh
```

In [68]:
```
# Positional arguements
print("Hello {0}, your current balance is {1}".format('Abinash',20000))
```

```
Hello Abinash, your current balance is 20000
```

In [69]:
```
print("Hello, {0} , Can I name you {1}".format('Abinash','Abhi'))
```

```
Hello, Abinash , Can I name you Abhi
```

In [71]:
```
# Keyword arguements
print("Hello {name}, your current balance is: {blc}".format(name='Samir',blc= 6000))
```

```
Hello Samir, your current balance is: 6000
```

In [72]:
```
print("Hey {name},your current balance is {blc}".format(name='Jyothi',blc=12000))
```

```
Hey Jyothi,your current balance is 12000
```

In [73]:
```
# Index()
# The index() method returns the index of a substring inside the string (if found). If
```

In [74]:
```
text= 'Python is blooming'
```

In [79]:
```
res = text.index('is')
res
```

Out[79]: 7

In [85]:
```
print(text.index('is',7,9))
```

7

In [86]:
```
print(text.index('oo'))
```

12

In [87]:
```
print(text.index('ing'))
```

15

In [88]:
```
# Isalnum()
# The isalnum() method returns True if all characters in the string are alphanumeric (
# string.isalnum()
```

In [89]:
```
name1='alpha123'
```

In [90]:
```
name1.isalnum()
```

Out[90]: True

In [91]:
```
name2='text 456'
```

In [92]:
```
name2.isalnum()
```

Out[92]: False

In [93]:
```
# Isalpha()
```

In [97]:
```
name3='Subhamastubhavah'
```

In [98]:
```
name3.isalpha()
```

Out[98]: True

In [99]:
```
name4='ayushman bhavah'
```

In [100]:
```
name4.isalpha()
```

Out[100]:
```
False
```

In [101]:
```
name5= 'ayushman123'
```

In [102]:
```
name5.isalpha()
```

Out[102]:
```
False
```

In [103]:
```
# isdecimal()
# The isdecimal() returns:
# True if all characters in the string are decimal characters.
# False if at least one character is not decimal character.
```

In [108]:
```
Q= '123456'
```

In [109]:
```
Q.isdecimal()
```

Out[109]:
```
True
```

In [110]:
```
P='subh1994'
```

In [111]:
```
P.isdecimal()
```

Out[111]:
```
False
```

In [112]:
```
R='asdf bh5689 0000'
```

In [113]:
```
R.isdecimal()
```

Out[113]:
```
False
```

In [114]:
```
# isdigit()
# The isdigit() method returns True if all characters in a string are digits. If not,
```

In [116]:
```
strng= '7891011121314'
```

In [117]:
```
strng.isdigit()
```

Out[117]:
```
True
```

In [118]:
```
strng1='Subhendu147'
```

In [120]:
```
strng1.isdigit()
```

Out[120]:
```
False
```

In [121]:
```
# isidentifier()
# The isidentifier() method returns:
# True if the string is a valid identifier
# False if the string is not a invalid identifier
```

In [123]:
```
name= 'Python3'
```

In [124]:
```
name.isidentifier()
```

Out[124]:
```
True
```

In [125]:
```
name1='python3456'
```

In [126]:
```
name1.isidentifier()
```

Out[126]:
```
True
```

In [131]:
```
name2='22python'
```

In [132]:
```
name2.isidentifier()
```

Out[132]:
```
False
```

In [133]:
```
name3='Py thon 3'
```

In [134]:
```
name3.isidentifier()
```

Out[134]:
```
False
```

In [135]:
```
# islower()
# The islower() method returns:
# True if all alphabets that exist in the string are lowercase alphabets.
# False if the string contains at least one uppercase alphabet.
```

In [136]:
```
dom= 'subhendu'
```

In [137]:
```
dom.islower()
```

Out[137]:
```
True
```

In [138]:
```
dom1='Subhendu'
```

In [139]:
```
dom1.islower()
```

Out[139]:
```
False
```

In [140]:
```
# isnumeric()
# The isnumeric() method checks if all the characters in the string are numeric.
```

In [143]:
```
dom2= '12000'
```

In [144]:
```
dom2.isnumeric()
```

Out[144]:
```
True
```

In [145]:
```
dom3= 'A56890'
```

In [146]:
```
dom3.isnumeric()
```

Out[146]:
```
False
```

In [148]:
```
# isprintable()
# The isprintable() method returns True if all characters in the string are printable.
```

In [150]:
```
sen4 = ' It is so easy to install'
```

In [151]:
```
sen4.isprintable()
```

Out[151]:
```
True
```

In [152]:
```
sen5= '                    '
```

In [153]:
```
sen5.isprintable()
```

Out[153]:
```
True
```

In [156]:
```
sen6= 'alpha_numeric\n/:'
```

In [157]:
```
sen6.isprintable()
```

Out[157]:
```
False
```

In [158]:
```
# isspace()
# isspace() method returns:
# True if all characters in the string are whitespace characters
# False if the string is empty or contains at least one non-printable character
```

In [159]:
```
k= '   klmnop'
```

In [160]:
```
k.isspace()
```

Out[160]:
```
False
```

In [161]:
```
l= '      '
```

In [162]:
```
l.isspace()
```

Out[162]:
```
True
```

In [163]:
```
# istitle()
# The istitle() method returns:
# True if the string is a titlecased string
# False if the string is not a titlecased string or an empty string
```

In [165]:
```
s='How Do You Feel?'
```

In [166]:
```
s.istitle()
```

Out[166]:
```
True
```

In [168]:
```
d='how are you?'
```

In [169]:
```
d.istitle()
```

Out[169]:
```
False
```

In [171]:
```
# isupper()
# The isupper() method returns:
# True if all characters in a string are uppercase characters
# False if any characters in a string are lowercase characters
```

In [172]:
```
f='WHEN SHOULD WE GO?'
```

In [173]:
```
f.isupper()
```

Out[173]:
```
True
```

In [174]:
```
f1="IT'S NOT Fare"
```

In [175]:
```
f1.isupper()
```

Out[175]:
```
False
```

In [1]:
```
# join()
# The string join() method returns a string by joining all the elements of an iterable
```

In [3]:
```
str= ['Python','is','fun']
```

In [11]:
```
print(' '.join(str))
```

```
Python is fun
```

In [12]:
```
str1= ['you','should','leave','now']
```

In [15]:
```
print('    '.join(str1))
```

```
you    should    leave    now
```

In [16]:
```
# ljust()
# ljust() method takes two parameters:

# width - width of the given string. If width is less than or equal to the length of t
# fillchar (Optional) - character to fill the remaining space of the width

# The ljust() method returns the left-justified string within the given minimum width.
# If fillchar is defined, it also fills the remaining space with the defined character
```

In [18]:
```
string= 'Anaconda'
```

In [19]:
```
print(string.ljust(15,'*'))
```

```
Anaconda*******
```

In [20]:
```
string1= 'School'
```

In [21]:
```
print(string1.ljust(12))
```

```
School
```

In [22]:
```
print(string.rjust(12))
```

```
    Anaconda
```

In [23]:
```
print(string1.rjust(15))
```

```
         School
```

In [24]:
```
# rjust()
# The rjust() method right aligns the string up to a given width using a specified cha
```

In [25]:
```
string2= 'jupyter'
```

In [26]:
```
print(string2.rjust(16,'.'))
```

```
.........jupyter
```

In [27]:
```
print(string2.rjust(16))
```

```
         jupyter
```

In [28]:
```
print(string2.rjust(16,'@'))
```

```
@@@@@@@@@jupyter
```

In [29]:
```
# lower()
# The lower() method converts all uppercase characters in a string into lowercase char
```

In [30]:
```
sen= 'PYTHON AND POWERBI ARE BLOOMING RIGHT NOW'
```

In [31]:
```
print(sen.lower())
```

```
python and powerbi are blooming right now
```

In [32]:
```
sen1='WHAT IS YOUR NAME'
```

In [33]:
```
print(sen1.lower())
```

```
what is your name
```

In [34]:
```
# upper()
# The upper() method converts all lowercase characters in a string into uppercase char
```

In [37]:
```
sen2= 'lets play badminton!'
```

In [38]:
```
print(sen2.upper())
```

```
LETS PLAY BADMINTON!
```

In [39]:
```
sen_2='when you will leave for home?'
```

In [40]:
```
print(sen_2.upper())
```

```
WHEN YOU WILL LEAVE FOR HOME?
```

In [41]:
```
# swapcase()
# The swapcase() method returns the string by converting all the characters to their o
```

In [42]:
```
sen3='cOmE ON, lEtS Go to LibRArY! '
```

In [43]:
```
print(sen3.swapcase())
```

```
CoMe on, LeTs gO TO lIBraRy!
```

In [44]:
```
sen_3='WhAt HAppEnED, WhY ARe yOU sO AnGRy!'
```

In [45]:
```
print(sen_3.swapcase())
```

```
wHaT haPPeNed, wHy arE You So aNgrY!
```

In [46]:
```
# lstrip()
# The lstrip() removes characters from the left based on the argument (a string specif
# removing leading spaces
```

In [47]:
```
sen4='     it could be way better! ****'
```

In [48]:
```
print(sen4.lstrip())
```

```
it could be way better! ****
```

In [49]:
```
sen_4='        **** It could be much better! ****'
```

In [50]:
```
print(sen_4.lstrip())
```

```
**** It could be much better! ****
```

In [51]:
```
# rstrip()
# The rstrip() method returns a copy of the string with trailing characters removed (b
# removing trailing spaces
```

In [52]:
```
sen5= 'hey,how are you!         '
```

In [53]:
```
print(sen5.rstrip())
```

```
hey,how are you!
```

In [54]:
```
sen_5="hey,how have you been! ****      "
```

In [55]:
```
print(sen_5.rstrip())
```

```
hey,how have you been! ****
```

In [56]:
```
# strip()
# The strip() method returns a copy of the string by removing both the leading and the
```

In [57]:
```
mes= '    We have to learn first for data science       '
```

In [58]:
```
print(mes.strip())
```

```
We have to learn first for data science
```

In [59]:
```
mes1='   how irritating the kiddo is    '
```

In [60]:
```
print(mes1.strip())
```

```
how irritating the kiddo is
```

In [61]:
```
# partition()
# The partition() method takes a string parameter separator that separates the string
```

In [63]:
```
strin= 'What should I do now'
```

In [65]:
```
print(strin.partition('should'))
```

```
('What ', 'should', ' I do now')
```

In [66]:
```
print(strin.partition('do'))
```

```
('What should I ', 'do', ' now')
```

In [67]:
```
print(strin.partition('what'))
```

```
('What should I do now', '', '')
```

In [68]:
```
# maketrans()
# The maketrans() method returns a translation table with a 1-to-1 mapping of a Unicod
```

In [86]:
```
dict= {"d":"123","e":"134","f":"389"}
string="abc"
```

In [87]:
```
print(string.maketrans(dict))
```

```
{100: '123', 101: '134', 102: '389'}
```

In [94]:
```
dict1={"k":"345","l":"456","m":"789"}
string1="abc"
```

In [95]:
```
print(string.maketrans(dict1))
```

```
{107: '345', 108: '456', 109: '789'}
```

In [96]:
```
# rpartition()
# rpartition() method takes a string parameter separator that separates the string at
```

In [157]:
```
strn= 'I am glad that you came'
```

In [158]:
```
strn[0]
```

Out[158]:
```
'I'
```

In [161]:
```
strn_1=(strn.rpartition('that'))
```

In [162]:
```
strn_1[0]
```

Out[162]:
```
'I am glad '
```

In [100]:
```
print(strn.rpartition('glad'))
```

```
('I am ', 'glad', ' that you came')
```

In [101]:
```
# translate()
# translate() method takes the translation table to replace/translate characters in th
```

In [108]:
```
zen='abc'
zen2='xyz'
ZEN_original='abcdef'
```

In [109]:
```
table=ZEN_original.maketrans(zen,zen2)
```

In [110]:
```
print("translated zen: ",zen.translate(table))
```

```
translated zen:  xyz
```

In [111]:
```
# replace()
# The replace() method replaces each matching occurrence of a substring with another s
```

In [112]:
```
song='ram ram jai raja ram'
```

In [113]:
```
print(song.replace('raja','sita'))
```

```
ram ram jai sita ram
```

In [114]:
```
song1='jai hanuman gyan gun sagar'
```

In [115]:
```
print(song1.replace('hanuman','pingakshaya'))
```

```
jai pingakshaya gyan gun sagar
```

In [116]:
```
# rfind()
# rfind() method returns an integer value.

# If substring exists inside the string, it returns the highest index where substring
# If substring doesn't exist inside the string, it returns -1.
```

In [117]:
```
Quote='He has completed his dummy project'
```

In [118]:
```
print(Quote.rfind('completed'))
```

```
7
```

In [119]:
```
print(Quote.rfind('get'))
```

```
-1
```

In [120]:
```
print(Quote.rfind('dummy'))
```

```
21
```

In [121]:
```
# rindex()
# If substring exists inside the string, it returns the highest index in the string wh
# If substring doesn't exist inside the string, it raises a ValueError exception.
# rindex() method is similar to rfind() method for strings.
```

In [122]:
```
quote= 'She went to parlour for a refresh'
```

In [124]:
```
print(quote.rindex('went'))
```

```
4
```

In [125]:
```
print(quote.rindex('refresh'))
```

```
26
```

In [126]:
```
# split()
# The split() method splits a string at the specified separator and returns a list of
```

In [135]:
```
quote_1= 'animal-fun-gun-sun'
```

In [136]:
```
print(quote_1.split('-'))
```

```
['animal', 'fun', 'gun', 'sun']
```

In [138]:
```
quote_2='what do you do for living?'
```

In [139]:
```
print(quote_2.split(' '))
```

```
['what', 'do', 'you', 'do', 'for', 'living?']
```

In [140]:
```
# rsplit()
# rsplit() breaks the string at the separator starting from the right and returns a li
```

In [143]:
```
mess= 'Respect your elders!'
```

In [144]:
```
print(mess.rsplit(' '))
```

```
['Respect', 'your', 'elders!']
```

In [145]:
```
mess1= 'Love your youngers!'
```

In [146]:
```
print(mess1.rsplit(' '))
```

```
['Love', 'your', 'youngers!']
```

In [149]:
```
mess2= 'bus_car_truck_auto_rikshaw_ghost'
```

In [150]:
```
print(mess2.rsplit('_'))
```

```
['bus', 'car', 'truck', 'auto', 'rikshaw', 'ghost']
```

In [151]:
```
# splitlines()
# The splitlines() method returns:

# a list of lines in the string.
# If there are not line break characters, it returns a list with a single item (a sing
```

In [152]:
```
line='Om\nNamo\nBhagavate\nBasudevaya'
```

In [154]:
```
print(line.splitlines())
```

```
['Om', 'Namo', 'Bhagavate', 'Basudevaya']
```

In [155]:
```
line1= '''Yadevi
        sarvabhuteshu
        shakti
        rupen
        samsthitah'''
```

In [156]:
```
print(line1.splitlines())
```

```
['Yadevi', '        sarvabhuteshu', '        shakti', '        rupen', '        samsth
itah']
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [7]:
```
def my_func(name,ph,state="karnataka"):
    print("username: ",name)
    print("phone: ",ph)
    print("state: ",state)
```

In [8]:
```
my_func("subh",1234)
```

```
username:  subh
phone:  1234
state:  karnataka
```

In [16]:
```
def func_2():
    for i in range(1,6):
        print(i)
    range(1,6)
```

In [17]:
```
func_2()
```

```
1
2
3
4
5
```

In [ ]:
```

```