String help

print(help(str))

1)

Using \_\_add\_\_:

val\_1="Hello"  
val\_2="There"  
val\_3=val\_1.\_\_add\_\_(val\_2)  
print(val\_3)

output: HelloThere

2)Capitalize

val\_1="hello"  
val\_2="There"  
val\_3=val\_1.capitalize()  
print(val\_3)

output:

Hello

3)casefold()

val\_1="Hello"  
val\_2="There"  
val\_3=val\_1.casefold()  
print(val\_3)

3)center()

val\_1="Hello Brother"  
val\_2="There"  
val\_3=val\_1.center(20)  
print(val\_3)

output:

Hello Brother

4)count()

val\_1="I love apples, apple are my favorite fruit"  
val\_2="There"  
val\_3=val\_1.count("apple")  
print(val\_3)

output:

2

5)encode()

The encode() method encodes the string, using the specified encoding. If no encoding is specified, UTF-8 will be used.

txt = "My name is Ståle S\_ut#"  
  
x = txt.encode()  
  
print(x)

output:

b'My name is St\xc3\xa5le S\_ut#'

6) endswith()

txt = "My name is Ståle S\_ut#"  
  
x = txt.endswith("#")  
  
print(x)

output:

True

7)expandtabs()

The expandtabs() method sets the tab size to the specified number of whitespaces.

txt = "H\te\tl\tl\to"  
  
x = txt.expandtabs(5)  
  
print(x)

Output:

H e l l o

8)find()

The find() method finds the first occurrence of the specified value.

The find() method returns -1 if the value is not found.

The find() method is almost the same as the [index()](https://www.w3schools.com/python/ref_string_index.asp) method, the only difference is that the index() method raises an exception if the value is not found. (See example below)

txt = "Hello, welcome to my world."  
  
x = txt.find("welcome")  
  
print(x)

Output:

7

9)format()

The format() method formats the specified value(s) and insert them inside the string's placeholder.

The placeholder is defined using curly brackets: {}. Read more about the placeholders in the Placeholder section below.

The format() method returns the formatted string.

txt = "For only {price:.2f} dollars!"  
print(txt.format(price = 49))

output:

For only 49.00 dollars!

10) index()

txt = "Hello, welcome to my world."  
  
x = txt.index("welcome")  
  
print(x)

output: 7

11)isalnum()

txt = "Company12" #here all are alpha numeric (0-9) and (A-Z)  
txt\_1="\*"  
  
x = txt.isalnum()  
y=txt\_1.isalnum()  
  
print(x)  
print(y)

output:

True

False

12)isalpha()

txt = "CompanyX" #all in alphabet  
  
x = txt.isalpha()  
  
print(x)

output:

True

13)isdecimal()

txt = "\u0033" #unicode for 3  
txt\_1="345"  
x = txt.isdecimal()  
y = txt\_1.isdecimal()  
print(x)  
print(y)

output:

True

True

14)isdigit()

txt = "50800"  
  
x = txt.isdigit()  
  
print(x)

Output:

True

15)isidentifier()

*'''The isidentifier() method returns True if the string is a valid identifier, otherwise False.  
  
A string is considered a valid identifier if it only contains alphanumeric letters (a-z) and (0-9), or underscores (\_). A valid identifier cannot start with a number, or contain any spaces.'''*txt = "Demo"  
  
x = txt.isidentifier()  
  
print(x)

Output:

True

16) islower()

*'''The islower() method returns True if all the characters are in lower case, otherwise False.  
  
Numbers, symbols and spaces are not checked, only alphabet characters.'''*txt = "hello world!"  
  
x = txt.islower()  
  
print(x)

Output: True

17)isupper()

*'''The isupper() method returns True if all the characters are in upper case, otherwise False.  
  
Numbers, symbols and spaces are not checked, only alphabet characters.'''*txt = "THIS IS NOW!"  
  
x = txt.isupper()  
  
print(x)

Output:

True

18)join()

*'''The join() method takes all items in an iterable and joins them into one string.  
  
A string must be specified as the separator.'''*myTuple = ("John", "Peter", "Vicky")  
  
x = "#".join(myTuple) #adding them with # as we added this  
  
print(x)

Output:

John#Peter#Vicky

19) ljust()

*'''The ljust() method will left align the string, using a specified character (space is default) as the fill character.'''*txt = "banana"  
  
x = txt.ljust(20) #will add 20 space before anoher strings  
  
print(x, "is my favorite fruit.")

Output:

banana is my favorite fruit.

20)lower()

*'''The lower() method returns a string where all characters are lower case.  
  
 Symbols and Numbers are ignored.'''*txt = "Hello my FRIENDS"  
  
x = txt.lower()  
  
print(x)

Output:

hello my friends

21)lstrip()

*'''The lstrip() method removes any leading characters (space is the default leading character to remove)'''*txt = ",,,,,ssaaww.....banana"  
  
x = txt.lstrip(",.asw") #removes , . a s w  
  
print(x)

Output:

Banana

22)maketrans()

*'''The maketrans() method returns a mapping table that can be used with the translate() method to replace specified characters.'''*txt = "Hi Sam!"  
x = "mSa"  
y = "eJo"  
mytable = txt.maketrans(x, y)  
print(txt.translate(mytable)) #replacing using mapping table

Output:

Hi Joe!

23)

Partition()

*'''The partition() method searches for a specified string, and splits the string into a tuple containing three elements.  
  
The first element contains the part before the specified string.  
  
The second element contains the specified string.  
  
The third element contains the part after the string.'''*txt = "I could eat bananas all day"  
  
x = txt.partition("bananas") #making a tuple basis on bananas  
  
print(x)

Output:

('I could eat ', 'bananas', ' all day')

24)replace()

txt = "one one was a race horse, two two was one too."  
  
x = txt.replace("one", "three")  
  
print(x)

output:

three three was a race horse, two two was three too.

25)rfind()

*'''The rfind() method finds the last occurrence of the specified value.  
  
The rfind() method returns -1 if the value is not found.  
  
The rfind() method is almost the same as the rindex() method. See example below.'''*txt = "Mi casa, su casa."  
  
x = txt.rfind("casa") #last casa started from 12th index  
  
print(x)

Output:

12

26)rindex()

txt = "Hello, welcome to my world."  
  
x = txt.rindex("e")#last e 's index  
  
print(x)

output:

12

27)rjust()

txt = "banana"  
  
x = txt.rjust(20, "O")#using 0 in the 20 spaces in right side  
  
print(x)

output:

OOOOOOOOOOOOOObanana

28)rpartition()

*'''The rpartition() method searches for the last occurrence of a specified string, and splits the string into a tuple containing three elements.  
  
The first element contains the part before the specified string.  
  
The second element contains the specified string.  
  
The third element contains the part after the string.'''*txt = "I could eat bananas all day, bananas are my favorite fruit"  
  
x = txt.rpartition("bananas")#making 3 element although the searched object ("bananas" in this code) is found or not  
  
print(x)

Output:

('I could eat bananas all day, ', 'bananas', ' are my favorite fruit')

29)

Syntax

*string*.rsplit(*separator, maxsplit*)

Parameter Values

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *separator* | Optional. Specifies the separator to use when splitting the string. By default any whitespace is a separator |
| *maxsplit* | Optional. Specifies how many splits to do. Default value is -1, which is "all occurrences" |

rsplit()

txt = "apple, banana, cherry, royen, faru"  
  
# setting the maxsplit parameter to 1, will return a list with 2 elements!  
x = txt.rsplit(", ",1)# here 1 means that there will be only 2 elements created and specially the the last one will always be an element and others will be there for this case  
  
print(x)

output:

['apple, banana, cherry, royen', 'faru']

txt = "apple, banana, cherry, royen, faru"  
  
# setting the maxsplit parameter to 1, will return a list with 2 elements!  
x = txt.rsplit(", ",2)  
  
print(x)

output:

['apple, banana, cherry', 'royen', 'faru']

30)

split()

txt = "hello, my name is Peter, I am 26 years old"  
  
x = txt.split(", ")#split on basis of ", "  
  
print(x)

output:

['hello', 'my name is Peter', 'I am 26 years old']

txt = "hello, my name is Peter, I am 26 years old"  
  
x = txt.split("my")#split on basis of ", "  
  
print(x)

output:

['hello, ', ' name is Peter, I am 26 years old']

31)splitlines()

*'''The splitlines() method splits a string into a list. The splitting is done at line breaks.'''*txt = "Thank you for the music\nWelcome to the jungle"  
  
x = txt.splitlines(True)#splits the string into a list and based on line break \n  
  
print(x)

Output:

['Thank you for the music\n', 'Welcome to the jungle']

32)startswith()

txt = "Hello, welcome to my world."  
  
x = txt.startswith("wel", 7, 20)#Check if position 7 to 20 starts with the characters "wel":  
  
print(x)

output:

True

33)strip()

txt = ",,,,,rrttgg.....banana....rrr"  
  
x = txt.strip(",.grt")#removes all ".grt" from both the sides  
  
print(x)

output:

banana

34)swapcase()

txt = "Hello My Name Is PETER"  
  
x = txt.swapcase() # making capital to lower and lower to capital  
  
print(x)

output:

hELLO mY nAME iS peter

35)title()

*'''The title() method returns a string where the first character in every word is upper case. Like a header, or a title.  
  
If the word contains a number or a symbol, the first letter after that will be converted to upper case.'''*txt = "Welcome to my 2nd world"  
  
x = txt.title()#making first character of every string in upper case  
  
print(x)

Output:

Welcome To My 2Nd World

36)translate()

#use a dictionary with ascii codes to replace 83 (S) with 80 (P):  
mydict = {83: 80}#Replace any "S" characters with a "P" character:  
txt = "Hello Sam!"  
print(txt.translate(mydict))

Output:

Hello Pam!

37)zfill()

a = "hello"  
b = "welcome to the jungle"  
c = "10.000"  
  
print(a.zfill(10))#Fill the strings with zeros until they are 10 characters long:  
print(b.zfill(10))#Fill the strings with zeros until they are 10 characters long:  
print(c.zfill(10))#Fill the strings with zeros until they are 10 characters long:

output:

00000hello

welcome to the jungle

000010.000