String Functions:

**string.isalpha()**

**Parameters:**

isalpha() does not take any parameters

**Returns :**

1.True- If all characters in the string are alphabet.

2.False- If the string contains 1 or more non-alphabets.

string = 'Ayush'

print(string.isalpha())

string = 'Ayush0212'

print(string.isalpha())

# checking if space is an alphabet

string = 'Ayush Saxena'

print( string.isalpha())

2. isalnum() function in Python programming language checks whether all the characters in a given string is alphanumeric or not.

**Alphanumeric**:A character that is either a letter or a number

**Return:**

***True:****If all the characters are alphanumeric****False:****If one or more characters are not alphanumeric*

Example :

string = "abc123"

print(string.isalnum())

# here a,b and c are characters and 1,2 and 3

# are numbers but space is not a alphanumeric

# character

string = "abc 123"

print(string.isalnum())

| **Method** | **True if** |
| --- | --- |
| str.isalnum() | String consists of only alphanumeric characters (no symbols) |
| str.isalpha() | String consists of only alphabetic characters (no symbols) |
| str.islower() | String’s alphabetic characters are all lower case |
| str.isnumeric() | String consists of only numeric characters |
| str.isspace() | String consists of only whitespace characters |
| str.istitle() | String is in title case |
| str.isupper() | String’s alphabetic characters are all upper case |

Let’s look at a couple of these in action:

number = "5"

letters = "abcdef"

print(number.isnumeric())

print(letters.isnumeric())

Example:

movie = "2001: A SAMMY ODYSSEY"

book = "A Thousand Splendid Sharks"

poem = "sammy lived in a pretty how town"

print(movie.islower())

print(movie.isupper())

print(book.istitle())

print(book.isupper())

print(poem.istitle())

print(poem.islower())

Example:

open\_source = "Sammy contributes to open source."

print(len(open\_source))

**join(), split(), and replace() Methods**

The str.join(), str.split(), and str.replace() methods are a few additional ways to manipulate strings in Python.

The str.join() method will concatenate two strings, but in a way that passes one string through another.

Let’s create a string:

balloon = "Sammy has a balloon."

Copy

Now, let’s use the str.join() method to add whitespace to that string, which we can do like so:

" ".join(balloon)

Copy

If we print this out:

print(" ".join(balloon))

Copy

We will see that in the new string that is returned there is added space throughout the first string:

Ouput

S a m m y h a s a b a l l o o n .

We can also use the str.join() method to return a string that is a reversal from the original string:

print("".join(reversed(balloon)))

Copy

Ouput

.noollab a sah ymmaS

We did not want to add any part of another string to the first string, so we kept the quotation marks touching with no space in between.

The str.join() method is also useful to combine a list of strings into a new single string.

Let’s create a comma-separated string from a list of strings:

print(",".join(["sharks", "crustaceans", "plankton"]))

Copy

Ouput

sharks,crustaceans,plankton

If we want to add a comma and a space between string values in our new string, we can simply rewrite our expression with a whitespace after the comma: ", ".join(["sharks", "crustaceans", "plankton"]).

Just as we can join strings together, we can also split strings up. To do this, we will use the str.split() method:

print(balloon.split())

Copy

Ouput

['Sammy', 'has', 'a', 'balloon.']

The str.split() method returns a list of strings that are separated by whitespace if no other parameter is given.

We can also use str.split() to remove certain parts of an original string. For example, let’s remove the letter a from the string:

print(balloon.split("a"))

Copy

Ouput

['S', 'mmy h', 's ', ' b', 'lloon.']

Now the letter a has been removed and the strings have been separated where each instance of the letter a had been, with whitespace retained.

The str.replace() method can take an original string and return an updated string with some replacement.

Let’s say that the balloon that Sammy had is lost. Since Sammy no longer has this balloon, we will change the substring "has" from the original string balloon to "had" in a new string:

print(balloon.replace("has","had"))

Within the parentheses, the first substring is what we want to be replaced, and the second substring is what we are replacing that first substring with. Our output will look like this:

Ouput

Sammy had a balloon.

Using the string methods str.join(), str.split(), and str.replace() will provide you with greater control to manipulate strings in Python.

Example:

def findDigitsCharsSymbols(inputString):

words = inputString.split()

charCount = 0

digitCount = 0

symbolCount = 0

for char in inputString:

if char.islower() or char.isupper():

charCount+=1

elif char.isnumeric():

digitCount+=1

else:

symbolCount+=1

print("Chars = ", charCount, "Digits = ", digitCount, "Symbol = ", symbolCount)

inputString = "P@#yn26at^&i5ve"

print("total counts of chars, digits,and symbols")

print(findDigitsCharsSymbols(inputString))

#### arrange String characters such that lowercase letters should come first

Given input String of combination of the lower and upper case arrange characters in such a way that all lowercase letters should come first.

Expected Output:

input\_String = PyNaTive

arranging characters giving precedence to lowercase letters

aeiNPTvy

arranging characters giving precedence to lowercase letters:

yaivePNT

inputStr = "PyNaTive"

words = inputStr.split()

lower = []

upper = []

for char in inputStr:

if char.islower():

lower.append(char)

else:

upper.append(char)

sortedString = ''.join(lower + upper)

print("\n arranging characters giving precedence to lowercase letters:")

print(sortedString)