Strings

In python, strings are sequence of characters and are created by enclosing either single quotation marks or double quotation marks. Python does not support character data type.

Example:

```
s = 'Python'        or        s = "Python"
```

The data type of string is implemented as an object of '**str**' class in python. The output of **type(s)** gives us **<class 'str'>**.

Indexing

Strings can be indexed (subscripted). Each character of string can be accessed using an index value, which starts at 0 and increments up to n-1 where n is length of the string. Indices may also be negative numbers, to start counting from the right.

P	y	t	h	O	n
0	1	2	3	4	5

```
s[0] → 'P'          s[3] → 'h'          s[5] → 'n'
s[-1] → 'n'         s[-6] → 'P'
```

Strings are immutable, that means, we *cannot modify the string contents*. In the above example, if we try to update **s[2]='T'**, the interpreter throws error. However, we can re-initialize string object **s** to hold new value, such as, **s='Java'**.

Slicing

We can extract a sub string from a string by providing a range of index values within a pair of square brackets.

Following are some slicing example:

```
s = 'Python'
```

Slicing expression	Meaning	Output
`s[1:4]`	Characters from index position 1 (included) to 4 (excluded)	'yth'
`s[:3]`	Characters from the beginning to index position 3 (excluded)	'Pyt'
`s[2:]`	Characters from index position 2 (included) to the end	'thon'
`s[-5:-2]`	Characters in reverse from fifth character (included) to second character (excluded)	'yth'
`s[-2:]`	Characters from the second-last (included) to the end	'on'
`s[:-4]`	Characters from beginning to fourth-last (excluded)	'Py'

String Methods

The `str` class has following methods that can be applied on string objects.

```
s = 'PYTHON is programmer FRIENDLY'
```

Method	Meaning	Example	Return value
index()	Returns the index of specified value (first occurrence, if many).	`s.index('N')`	5
isalpha()	Returns True, if the string contains alphabets (only)	`s.isalpha()`	False (as string contains space)
isdigit()	Returns True, if the string contains digits (only)	`s.isdigit()`	False
isalnum()	Returns True, if the string contains alphabets/digits	`s.isalnum()`	False
islower()	Returns True, if string is in lowercase	`s.islower()`	False
isupper()	Returns True, if string is in uppercase	`s.isupper()`	False
lower()	Converts to lowercase	`s.lower()`	'python is programmer friendly'
upper()	Converts to uppercase	`s.upper()`	'PYTHON IS PROGRAMMER FRIENDLY'
title()	Converts the first character of each word to upper case	`s.title()`	'Python Is Programmer Friendly'
capitalize()	Converts the first character to upper case	`s.capitalize()`	'Python is programmer friendly'
swapcase()	Swaps cases, lower case becomes upper case and vice versa	`s.swapcase()`	'python IS PROGRAMMER friendly'
count()	Returns the number of times a specified value occurs in a string	`s.count('r')`	3
replace()	Returns a string where a specified value is replaced with a	`s.replace('programmer', 'user')`	'PYTHON is user FRIENDLY'

	specified value		
split()	Splits the string at the specified separator, and returns a ***list***	`s.split(' ')`	['PYTHON', 'is', 'programmer', 'FRIENDLY'] *Note that a space is passed as argument to split() method.*

Other Library Functions

The basic functions of python such as **`len(), max(), min, sorted()`** can also be applied to strings. Consider the following examples:

```
s = 'aAbB19'
```

Function	Meaning	Example	Return value
len()	It accepts string object as argument and returns the length of specified string.	len(s)	6
max()	It accepts string object as argument and return the largest item *(based on the ASCII value)*.	max(s)	'b'
min()	It accepts string object as argument and return the smallest item *(based on the ASCII value)*.	min(s)	'1'
sorted()	It accepts string object as argument and returns a list containing each characters of string arranged in ascending order *(based on the ASCII value)*.	sorted(s)	['1', '9', 'A', 'B', 'a', 'b']