what is string

• A string is a data type in python.

• String is sequence of characters which are enclosed within a single quote or double quote. These characters could be anything like letters, numbers, or special symbols. For example, "python" is a string.

• A string is created by placing all the elements inside single ' ' or double " " or triple ''' ''' quotes.

• The string type is represented using a str class.

• String can be represented by using single ' ' or double " " or triple ''' ''' quotes.

• We can create empty string using string constructor str() or single quote ' '.

Ordered:-

it preserves sequence order.

Immutable:-

String is immutable data type this means that element of string can not be changed once created.

Duplicates:-

duplicate elements are allowed in string.

• It support homogenous and heterogenous data. Eg. 'abcd' '1234' 'a1b2'

• It is global data type which accept everything.

• To get the length of string we have len() function.

• The background data structure of string is array. Hence indexing is but obvious.

Indexing

Indexing is used for accessing single element. there are two types of indexing
1) positive indexing :- positive indexing start from 0 to n-1.
For example, suppose we have a string s = 'python' and if we want to access the 3rd element then need to use s[2] here we get 3rd element which is 't'.

2) negative indexing:- negative indexing start from -n to -1. For example, suppose we have a string s = 'python' and if we want to access the 3rd element then need to use s[-3] here we get 3rd element which is 't'. The index must be an integer. We can't use floats or other types, this will result into TypeError. If we mention the index value greater than the length of a string then it will throw an index error.

slicing

When we want access multiple elements in a sequence for this we have a slicing. Slicing is based on indexing only. We use : colon for slicing.
Specify the start index and the end index, separated by a colon, to return a part of the string.

Syntax: [start index : stop index : step] Eg. Suppose we have a string s = 'python' now here we can use slicing s[2:5] then we got 'tho'. Ending index is excluded.If we want reverse string then we can use [start index : stop index: -1]

basic quesions on string

In [1]:
```
# suppose we have a string
s = 'i love data science'
```

Que. How to open list of string methods?

In [2]:
```
# using dir(str_name) function we get all the methods of string.
dir(s)
```

Out[2]:
```
['__add__',
 '__class__',
 '__contains__',
 '__delattr__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getitem__',
 '__getnewargs__',
 '__gt__',
 '__hash__',
 '__init__',
 '__init_subclass__',
 '__iter__',
 '__le__',
 '__len__',
 '__lt__',
 '__mod__',
 '__mul__',
 '__ne__',
 '__new__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__rmod__',
 '__rmul__',
 '__setattr__',
 '__sizeof__',
 '__str__',
 '__subclasshook__',
 'capitalize',
 'casefold',
 'center',
 'count',
 'encode',
 'endswith',
 'expandtabs',
 'find',
 'format',
 'format_map',
 'index',
 'isalnum',
 'isalpha',
 'isascii',
 'isdecimal',
 'isdigit',
 'isidentifier',
 'islower',
 'isnumeric',
 'isprintable',
 'isspace',
 'istitle',
 'isupper',
 'join',
 'ljust',
 'lower',
 'lstrip',
 'maketrans',
 'partition',
 'removeprefix',
 'removesuffix',
 'replace',
 'rfind',
 'rindex',
 'rjust',
 'rpartition',
 'rsplit',
 'rstrip',
 'split',
 'splitlines',
 'startswith',
 'strip',
 'swapcase',
 'title',
 'translate',
 'upper',
 'zfill']
```

Que. How to get information about the particular method?

In [3]:
```
# using help(str.method_name) function
#Eg. How to know information about title()
help(s.title)
```

```
Help on built-in function title:

title() method of builtins.str instance
    Return a version of the string where each word is titlecased.

    More specifically, words start with uppercased characters and all remaining
    cased characters have lower case.
```

Que. How to get length of string?

In [4]:
```
# using len(str_name) function we get the length of the string
len(s)
```

Out[4]:
```
19
```

Que. How to create an empty string?

In [5]:
```
# python string is called an empty string if the string doesn't have any character, whitespace whatsoever.
# Using single quote ' ' or double quote " "  we can create an empty string.
# assign '' or "" to a variable to create an empty string
str = ''
str = ""
str
```

Out[5]:
```
''
```

In [6]:
```
# To check an empty string in python use the len() function,and if it returns zero that means the given string is empty.
len(str)
```

Out[6]:
```
0
```

Que. How to create a Blank string?

In [7]:
```
# if the string has single white space or multiple white spaces or tab spaces then it is a black string.
str = ' '
str
```

Out[7]:
```
' '
```

In [8]:
```
# if we check length of string
len(str)
```

Out[8]:
```
1
```

Que. How to access string item?

There are many ways to access string items. 1) Indexing :-using indexing we can access single element from a string. For example, suppose we have a string s = 'python' and if we want to access the 3rd element then need to use s[2] here we get 3rd element which is 't'.

2) Slicing:- using slicing we can access multiple elements from string. Eg. Suppose we have a string s = 'python' now here we can use slicing s[2:5] then we got 'tho'. Ending index is excluded

In [9]:
```
# lets take an example,suppose we have a string
s
```

Out[9]:
```
'i love data science'
```

In [10]:
```
# using indexing
s[3]
```

Out[10]:
```
'o'
```

In [11]:
```
# at 6th index we have blank space.
s[6]
```

Out[11]:
```
' '
```

In [12]:
```
# using slicing.
s[3:8]
```

Out[12]:
```
'ove d'
```

In [13]:
```
# if i want 'data'
s[7:11]
```

Out[13]:
```
'data'
```