DAILY DSA | DAY-3 | STRINGS|
-GOPALKRISHNA A

Sequences of characters are referred to as strings. Strings can be any length and can include any character such as letters, numbers, symbols, and whitespace (Spaces, tabs, new lines)

In this post, we will quickly go through common operations in strings. Functions that are used in arrays can also be applied to strings

Operation	Time complexity
Access	O(1)
Search	O(n)
Insert	O(n)
Remove	O(n)

Operation	Time complexity
Find substring	O(1)
Concatenating strings	O(n + m)
Slice	O(m)
Split (by token)	O(n + m)
Strip (Remove leading & trailing whitespaces)	O(n)

we assume the other string is of length m

Operations of strings:

Escaping characters: Backlashes (\) is used to escape characters in a Python string.

```
txt = "Daily \"Data structures \"."
print(txt)
#Output: Daily "Data structures ".
```

The in syntax: The in syntax is used to determine if a letter or a substring exists in a string. It returns True if a match is found, otherwise, False is returned.

```
txt = "Daily data structures practise"

print("d" in txt) #Prints: True
print("x" in txt) #Prints: False
```

Indexing and Slicing strings: In Python, Strings can be indexed using the same notation as lists, since strings are lists of characters. A single character can be accessed with a bracket **([index])**, or a substring can be accessed using slicing **([start:end])**

```
str = 'yellow'
str[1]     # => 'e'
str[-1]    # => 'w'
str[4:6]   # => 'ow'
str[:4]    # => 'yell'
str[-3:]   # => 'low'
```

Iterate string: To iterate through string in Python, "for...in" notation is used

```
str = "hello"
for c in str:
  print(c)

# h
# e
# l
# l
# o
```

Built-in Function len(): In Python, the built-in len() function can be used to determine the length of an object, it can be used to compute the length of strings, lists, sets, and other countable objects.

```
length = len("Hello")
print(length)
# Output: 5

colors = ['red', 'yellow', 'green']
print(len(colors))
# Output: 3
```

String concatenation: To combine the content of two strings into a single string, Python provides the + Operator. This process of joining strings is called concatenation.

```
x = 'Data structures '
y = 'and algorithms'

z = x + y

print(z)
# Output: Data structures and
algorithms
```

Immutable strings: Strings are immutable in Python. This means that once a string has been defined, it can't be changed.
There are no mutating methods for strings. This is unlike data types like lists, which can be modified once they are created.

Index Error: When indexing into a string in Python, if you try to access an index that doesn't exist, an IndexError is generated.

```
txt = "Data"
index = txt[5]

Output: IndexError: string index out of
range
```

String method .strip(): The string method .strip() can be used to remove characters from the beginning and end of a string.

A string argument can be passed to the method, specifying the set of characters to be stripped. With no arguments to the method, white space is removed.

```
text = '    Data structures and
algorithms    '
print(text.strip())

Prints: Data structures and algorithms
```

Try removing (".") to strip the "." characters.

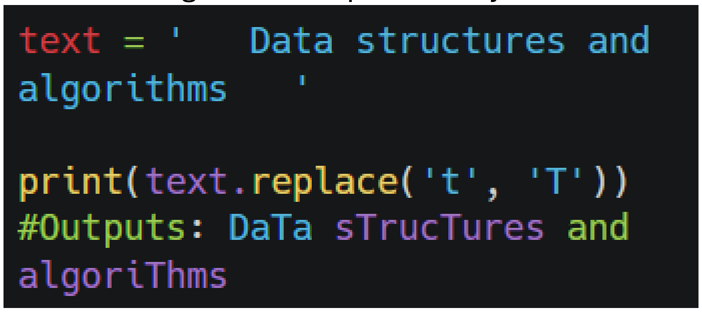
String method .split(): The string method .split() splits a string into a list of items:

- If no argument is passed, the default behavior is to split on whitespace
- If an argument is passed to the method, that value is used as the delimiter on which to split the string

```
text = '   Data structures and
algorithms   '

print(text.split())
#Output: ['Data', 'structures', 'and',
'algorithms']
print(text.split('t'))
#Output: ['   Da', 'a s', 'ruc', 'ures
and algori', 'hms   ']
```

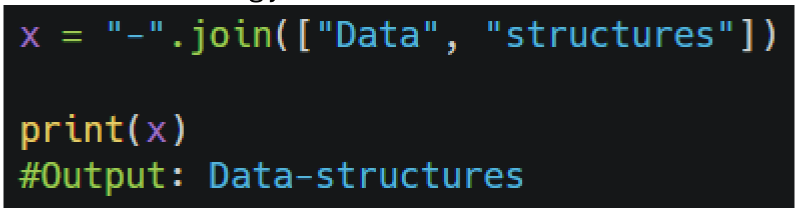
String replace: The .replace() method is used the replace the occurrence of the first argument with the second argument within the string.
The first argument is the old substring to be replaced, and the second argument is the new substring that will replace every occurrence of the first one with the string.



```
text = '   Data structures and
algorithms   '

print(text.replace('t', 'T'))
#Outputs: DaTa sTrucTures and
algoriThms
```

String method .join(): The string method .join() concatenates a list of strings together to create a new string joined with the desired delimiter



```
x = "-".join(["Data", "structures"])

print(x)
#Output: Data-structures
```

Misc operations:
string method .find(): The Python string method .find() returns the index of the first occurrence of the string passed as the argument. It returns -1 if no occurrence is found.

String method .upper(): The string method .upper() returns the string with all lowercase characters converted to uppercase.

String method .lower(): The string method .lower() returns a string with all uppercase characters converted into lowercase.

Practise problems from Geeks for Geeks on strings:
https://www.geeksforgeeks.org/string-data-structure/