Strings in Python

What is String?

- In Python, Strings are arrays of bytes representing Unicode characters. However, Python does not have a character data type, a single character is simply a string with a length of 1. Square brackets can be used to access elements of the string.

  Eg: name = “Alice”, name[0] = ‘A’,….

Assign String to a Variable

```
test_string = “Hello, World!”
print(“test_string”)
```

Multi-line Strings

```
multi_test = """"Hello, I am Alice.,
                    I live in Queens.,
                    I am 21 years old.""""
print(multi_test)
```

- You can use any of single or double quote.

Strings are Arrays

```
test_string = “Hello, World!”
print(test_string[4])
Output >> o
```

Looping Through a String

```
test_string = “Banana”

for i in test_string:
     print(i)

Output : >> Banana
```

Finding the length of a string

```
test_string = “I am a string.”

length = len(test_string)
print(length)
```

Output: >> 14

Checking Strings

- There are two important keywords to check for patterns or a word in [strings,...] in Python.
- They are :

  #in and #not in

- The #[in] keyword checks if the pattern is present in the string.
- The #[not in] keyword checks if the pattern is not present in the string.

Checking Strings

```
test_string = “You can learn and improve your
                    programming skills only by
                    practicing.”
if “improve” in test_string:
    print(“Pattern is found in the text!”)
Output: >> Pattern is found in the text!
```

- Try [not in] yourself

String Slicing

```
test_string = “Hello, world!”
print(test_string[2:]) # from pos_2 till end…
print(test_string[2:5])# output: >> llo [2, 3, 4]
print(test_string[:5]) # output: >> Hello
print(test_string[-5:-2]) #Output: >> orl
print(test_string[::-1])
```

- Find the output of the last line.

Modifying Strings

- Upper Case

  test_string = “hi”

  print(test_string.upper())

- Lower Case

  test_string = “HI”

  print(test_string.lower())

Modifying Strings

- Remove White-space
  - The strip() method removes any white-space from the beginning or the end from a string:

```
test_string = “ Hello, World! ”

print(test_string.strip())


Output: >> “Hello, World!”
```

Modifying Strings

- Replace String

  The replace() method replaces a string with another string:

  test_string = “Hello, World!”

  print(test_string.replace(“H”, “J”))

  #you can specify the “count” in the replace()

  Output: >> “Jello, World!”

Modifying Strings

- Splitting a String

  The split() method splits the string into sub-strings if it finds instances of the separator:

```
test_string = “Hello, World!”
print(test_string.split(“,”))
```

```
Output: >> [“Hello”, “ World!”]
```

String Concatenation

- String Concatenation

  To concatenate, or combine, two strings you can use the + operator.

```
test_str1 = “Hello”
test_str2 = “World”
print(test_str1 + test_str2)
Output: >> HelloWorld
```

Formatting strings

- String format

  The format() method takes the passed arguments, formats them, and places them in the string where the placeholders {} are:

```
name = “Alice”

temp = “I am {}”.format(name)

print(temp)
```

```
Output: >> “I am Alice.”
```

Formatting strings

- F – string

  To create an f-string, prefix the string with the letter “ f ”. The string itself can be formatted in much the same way that you would with str.format(). F-strings provide a concise and convenient way to embed python expressions inside string literals for formatting.

```
_name = “Alice”

print(f“I am {_name}.”)


Output: >> “I am Alice.”
```

Escape Characters

Code	Result
\’	Single Quote
\\	Back-slash
\n	New Line
\r	Carriage return
\t	Tab
\b	Back space
\f	Form feed
\ooo	Octal value
\xhh	Hex value

Methods of string

capitalize()	Converts the first character to upper case
casefold()	Converts string into lower case
center()	Returns a centered string
count()	Returns the number of times a specified value occurs in a string
encode()	Returns an encoded version of the string
endswith()	Returns true if the string ends with the specified value

Methods of string

expandtabs()	Sets the tab size of the string
find()	Searches the string for a specified value and returns the position of where it was found
format()	Formats specified values in a string
format_map()	Formats specified values in a string
index()	Searches the string for a specified value and returns the position of where it was found
isalnum()	Returns True if all characters in the string are alphanumeric

Methods of string

isalpha()	Returns True if all characters in the string are in the alphabet
isascii()	Returns True if all characters in the string are ascii characters
isdecimal()	Returns True if all characters in the string are decimals
isdigit()	Returns True if all characters in the string are digits
isidentifier()	Returns True if the string is an identifier
islower()	Returns True if all characters in the string are lower case

Methods of string

isnumeric()	Returns True if all characters in the string are numeric
isprintable()	Returns True if all characters in the string are printable
isspace()	Returns True if all characters in the string are whitespaces
istitle()	Returns True if the string follows the rules of a title
isupper()	Returns True if all characters in the string are upper case
join()	Converts the elements of an iterable into a string

Methods of string

ljust()	Returns a left justified version of the string
lower()	Converts a string into lower case
lstrip()	Returns a left trim version of the string
maketrans()	Returns a translation table to be used in translations
partition()	Returns a tuple where the string is parted into three parts
replace()	Returns a string where a specified value is replaced with a specified value

Methods of string

rfind()	Searches the string for a specified value and returns the last position of where it was found
rindex()	Searches the string for a specified value and returns the last position of where it was found
rjust()	Returns a right justified version of the string
rpartition()	Returns a tuple where the string is parted into three parts
rsplit()	Splits the string at the specified separator, and returns a list
rstrip()	Returns a right trim version of the string

Methods of string

split()	Splits the string at the specified separator, and returns a list
splitlines()	Splits the string at line breaks and returns a list
startswith()	Returns true if the string starts with the specified value
strip()	Returns a trimmed version of the string
swapcase()	Swaps cases, lower case becomes upper case and vice versa
title()	Converts the first character of each word to upper case

Methods of string

translate()	Returns a translated string
upper()	Converts a string into upper case
zfill()	Fills the string with a specified number of 0 values at the beginning

References

- https://www.w3schools.com/python/python_strings.asp
- https://www.tutorialspoint.com/python3/python_strings.htm

Thank You!