PYTHON

Page No.
Date

Strings

→ Strings in python are surrounded by either single quotation marks, or double quotation marks.
→ 'India' is the same as "India"

Create a variable containing STRING

```
a = "Python"
print(a)
```

⇒ Python

Multiline strings should enclosed within triple, single / double quotes.

Slicing

→ You can slice the string by passing its index.
→ Positive index start from ZERO and goes from left to right.
→ Negative index start from -1 and goes from right to left.
→ Starting index starts cutting slicing from same index.
→ Ending index ends slicing one step before.
→ Syntax: String [start: stop : step]

-6	-5	-4	-3	-2	-1
P	Y	T	H	O	N
0	1	2	3	4	5

Negative Indexing

→ You can use negative index in same as we used positive index.

Steps in Slicing

→ Steps decide the direction of printing for sliced index & how many steps.

→ Positive step means left to right & negative step means right to left.

→ Default step = 1, means take a 1 step from left to right.

→ Step = 2, means take 2 steps from right to left.

→ Approach should be as following:

1. Start slicing from StartIndex
2. End slicing at EndIndex - 1
3. After slicing between start & end index check step.
4. Print letter as per step sign & value.

String = "Hello World"

string [0:12:2]

⇒ 'HloWrd'

Page No.
Date

String Concatenation

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

→ If you want to add space then use syntax : a + " " + b

```
# Without space
a = "Hello"
b = "World"
c = a+b
print(c)
```

⇒ HelloWorld

```
# With space
a = "Hello"
b = "World"
c = a + " " + b
print(c)
```

⇒ Hello World

String Format

→ format() method takes the arguments in sequence & place them at placeholder {}.

→ If inside curly bracket, you are putting 1 like {2} then first passed argument will get placed at {2}.

```
a = "My Name Is {} and I am {} years old".format("Rahul", 21)
print(a)
```

⇒ My Name Is Rahul and I am 21 years old.

Escape Character

→ Escape character are used to escape illegal character in string.

→ An example of an illegal ~~char~~ character is a double quote inside a string that is surrounded by double quotes.

→ backlash "\" is used to escape the character.

Commonly used Escape character

Single Quote

```
string = 'Let\'s try.'
print (string)
```

⇒ Let's try.

Backlash

```
string = "Lets insert one \\ (backlash)."
print (string)
```

⇒ Lets insert one \ (backlash).

New Line

```
string = "Hello\nWould!"
print (string)
```

⇒ Hello
Would!

Tab

```
string = "Hello \t World!"
print (string)
```

⇒ Hello World!

POPULAR STRING METHODS

1.	capitalize()	converts the first character to upper case.
2.	casefold()	Converts string into lower case.
3.	center()	Returns a center string.
4.	count()	Returns the number of times a specified value occurs in a string.
5.	encode()	Returns an encoded version of the string.
6.	endswith()	Returns true if the string ends with the specified value.
7.	expandtabs()	Sets the tab size of the string.
8.	find()	Searches the string for a specified value & returns the position of where it was found.
9.	format()	Formats specified values in a string.
10.	format_map()	Formats specified values in a string.

11.	index()	Searches the string for a specified value & returns the position of where it was found.
12.	isalnum()	Returns True if all characters in the string are alphanumeric.
13.	isalpha()	Returns True if all characters in the string are alphabetic.
14.	isdecimal()	Returns True if all characters in the string are decimals.
15.	isdigit()	Returns True if all characters in the string are digits.
16.	isidentifier()	Returns True if all characters of the string is an identifier.
17.	islower()	Returns True if all characters in the string are lower case.
18.	isnumeric()	Returns True if all characters in the string are numeric.
19.	isprintable()	Returns True if all characters in the string are printable.
20.	isspace()	Returns True if all characters in the string are whitespaces.
21.	istitle()	Returns True if the string follows the rules of a title.
22.	isupper()	Returns True if all characters in the string are upper case.
23.	join()	Joins the elements of an iterable to the end of the string.

Date

24. ljust()	Returns a left justified version of the string.
25. lower()	Converts a string into lower case.
26. lstrip()	Returns a left trim version of the string
27. maketrans()	Returns a translation table to be used in translations.
28. partition()	Returns a tuple where the string is parted into three parts.
29. Replace()	Returns a string where a specified value is replaced with a specified value.
30. rfind()	Searches the string for a specified value and returns the last position of where it was found.
31. rindex()	Searches the string for a specified value and returns the last position of where it was found.

32.	rjust()	Returns a right justified version of the string.
33.	rpartition()	Returns a tuple where the string is parted into 3 parts.
34.	rsplit()	Splits the string at the specified separator, and returns a list.
35.	rstrip()	Returns a right trim version of the string.
36.	split()	Splits the string at the specified separator, and returns a list.
37.	splitlines()	Splits the string at line breaks and returns a list.
38.	startswith()	Returns true if the string starts with the specified value.
39.	strip()	Returns a trimmed version of string.
40.	swapcase()	Swaps cases, lower case becomes upper case and vice versa.

41.	title()	Converts the first character of each word to upper case.
42.	translate()	Returns a translated string.
43.	upper()	Converts a string into upper case.
44.	zfill()	Fills the string with a specified number of 0 values at the beginning.