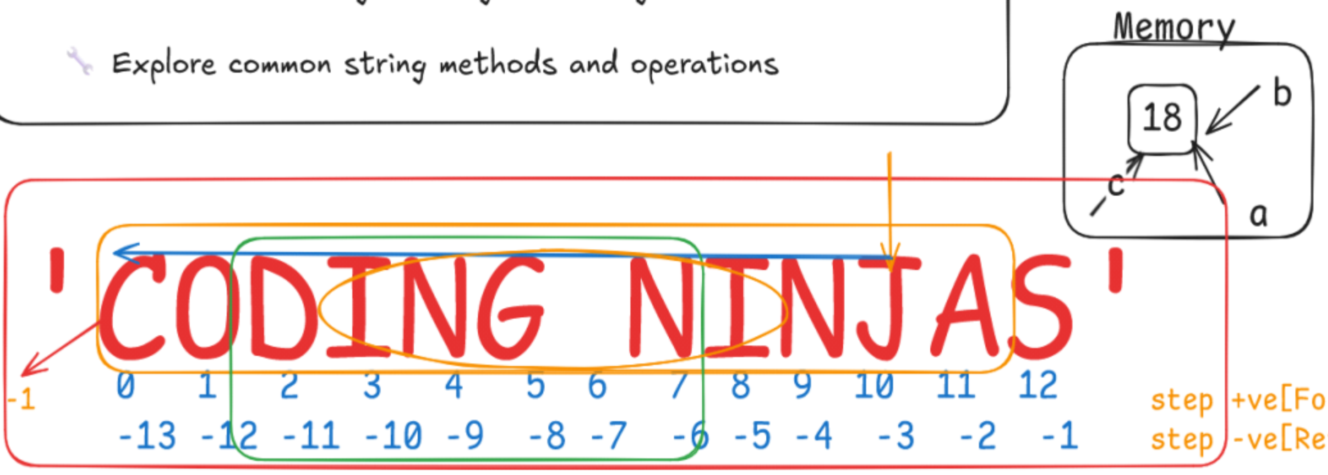
Strings

🎯 Session Objectives

- 🔤 Understand string indexing and slicing
- 🔧 Explore common string methods and operations



step +ve[Forward]
step -ve[Reverse]

Indexing [pos] -> return 'character' of that particular index.

Slicing[start:stop:step] -> return 'substring of a string'

Default : Start[0] , Stop [length of a string][Non-Inclusive] , Step[1]

```
_str = 'Coding Ninjas'
print(_str[:6]) # Coding
print(_str[7:]) # Stop[13 _ len] # Ninjas
```

```
print(_str[-11:7:1]) # 'ding '
print(_str[-5:-11:-2]) # 'i n'
```

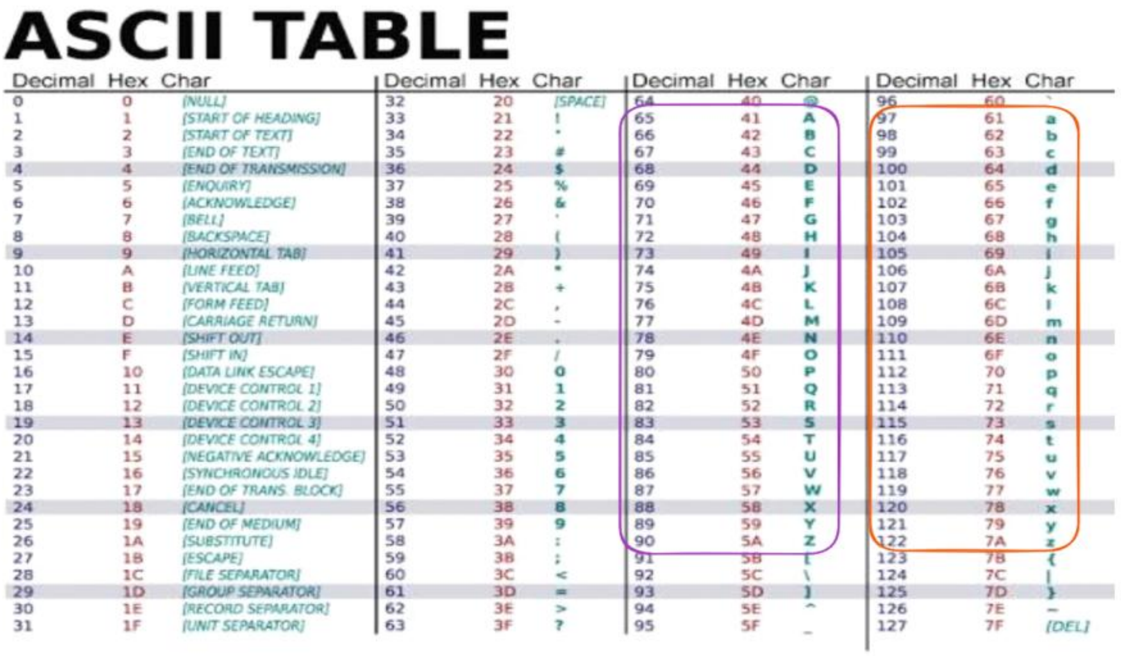
```
_str[-3:0] [Step : 1] ""

_str[-3:0:-1] 'jniN gnido'
```

ASCII TABLE

'a' > 'A'
'b' > 'B'

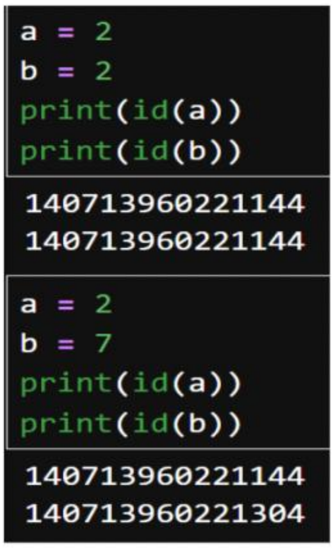
A[65]-Z[90] << a[97]-z[122]



ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Coding < coding - True

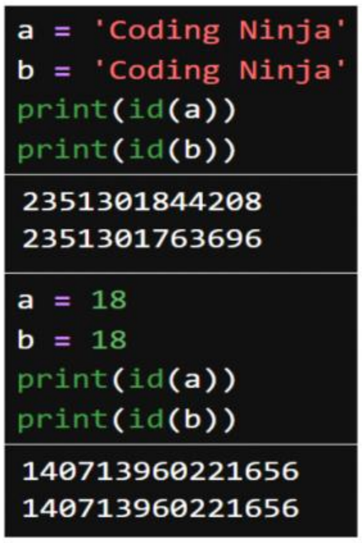


```
a = 2
b = 2
print(id(a))
print(id(b))
```

```
140713960221144
140713960221144
```

```
a = 2
b = 7
print(id(a))
print(id(b))
```

```
140713960221144
140713960221304
```

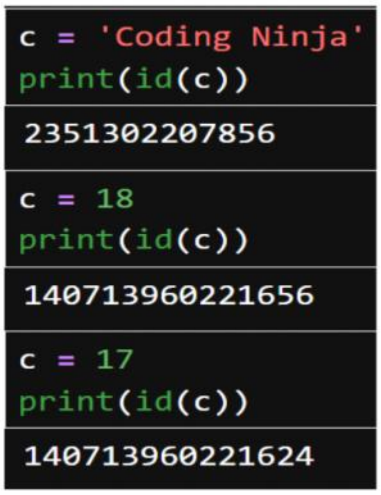


```
a = 'Coding Ninja'
b = 'Coding Ninja'
print(id(a))
print(id(b))
```

```
2351301844208
2351301763696
```

```
a = 18
b = 18
print(id(a))
print(id(b))
```

```
140713960221656
140713960221656
```



```
c = 'Coding Ninja'
print(id(c))
```

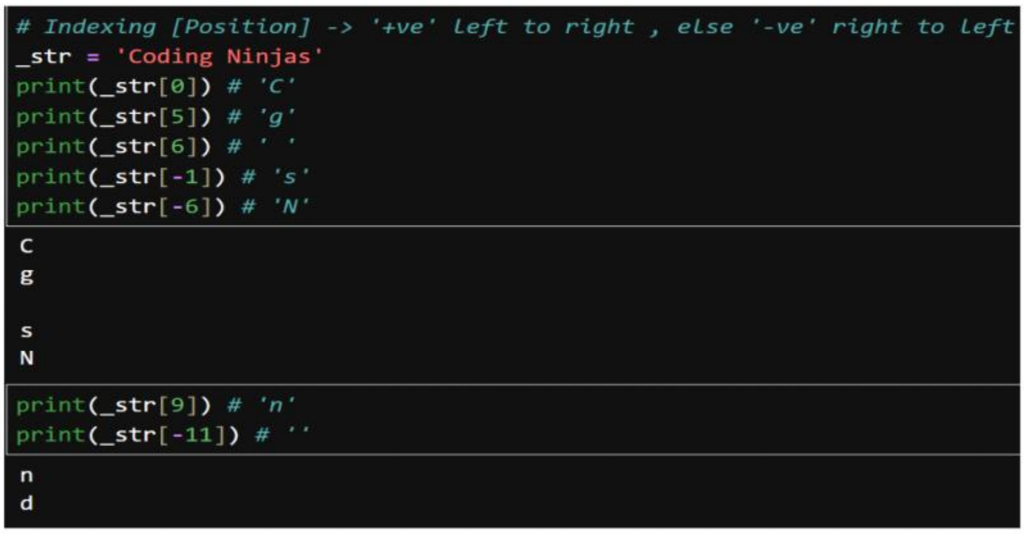
```
2351302207856
```

```
c = 18
print(id(c))
```

```
140713960221656
```

```
c = 17
print(id(c))
```

```
140713960221624
```



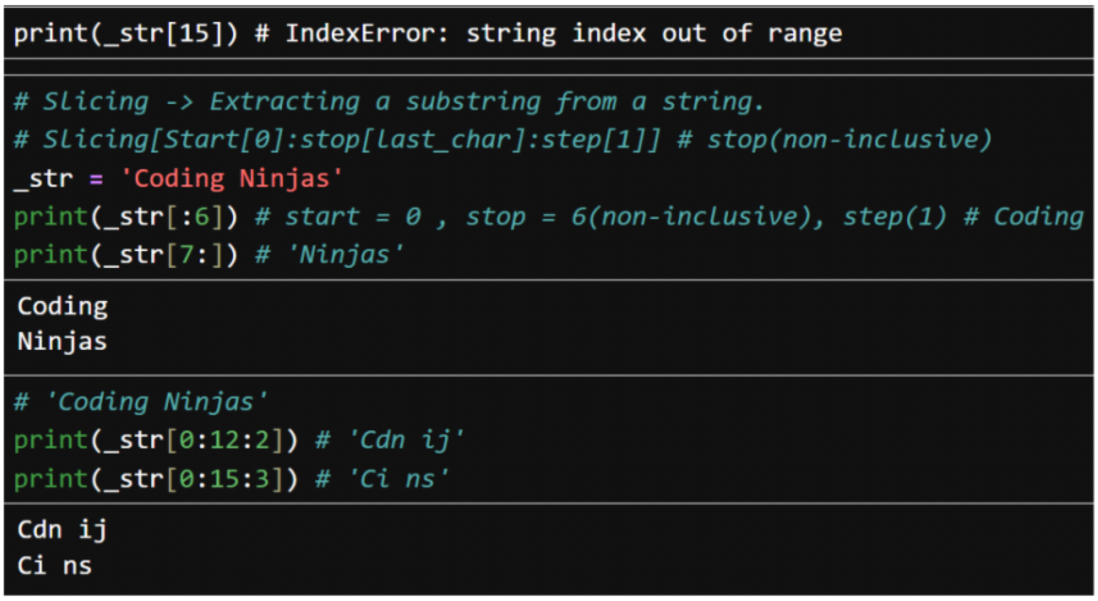
```
# Indexing [Position] -> '+ve' left to right , else '-ve' right to left
_str = 'Coding Ninjas'
print(_str[0]) # 'C'
print(_str[5]) # 'g'
print(_str[6]) # ' '
print(_str[-1]) # 's'
print(_str[-6]) # 'N'
```

```
C
g

s
N
```

```
print(_str[9]) # 'n'
print(_str[-11]) # ''
```

```
n
d
```



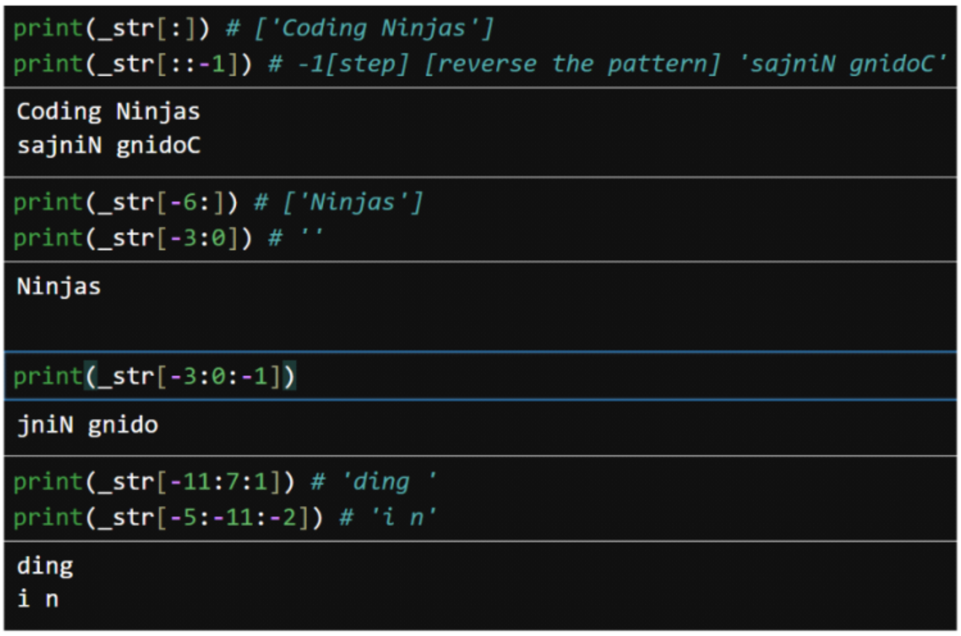
```
print(_str[15]) # IndexError: string index out of range
```

```
# Slicing -> Extracting a substring from a string.
# Slicing[Start[0]:stop[last_char]:step[1]] # stop(non-inclusive)
_str = 'Coding Ninjas'
print(_str[:6]) # start = 0 , stop = 6(non-inclusive), step(1) # Coding
print(_str[7:]) # 'Ninjas'
```

```
Coding
Ninjas
```

```
# 'Coding Ninjas'
print(_str[0:12:2]) # 'Cdn ij'
print(_str[0:15:3]) # 'Ci ns'
```

```
Cdn ij
Ci ns
```



```
print(_str[:]) # ['Coding Ninjas']
print(_str[::-1]) # -1[step] [reverse the pattern] 'sajniN gnidoC'
```

```
Coding Ninjas
sajniN gnidoC
```

```
print(_str[-6:]) # ['Ninjas']
print(_str[-3:0]) # ''
```

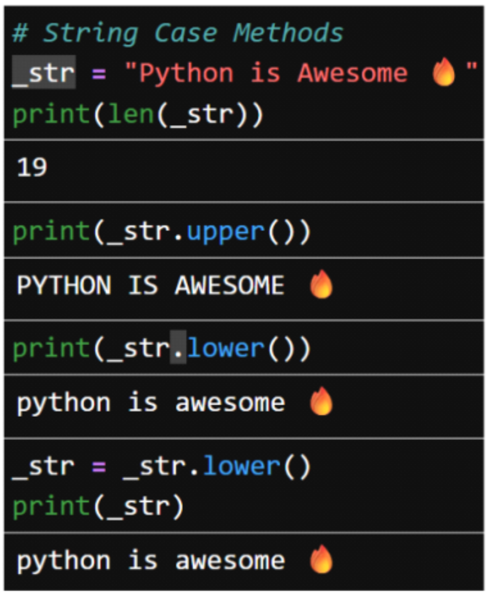
```
Ninjas
```

```
print(_str[-3:0:-1])
```

```
jniN gnido
```

```
print(_str[-11:7:1]) # 'ding '
print(_str[-5:-11:-2]) # 'i n'
```

```
ding
i n
```



```
# String Case Methods
_str = "Python is Awesome 🔥"
print(len(_str))
```

```
19
```

```
print(_str.upper())
```

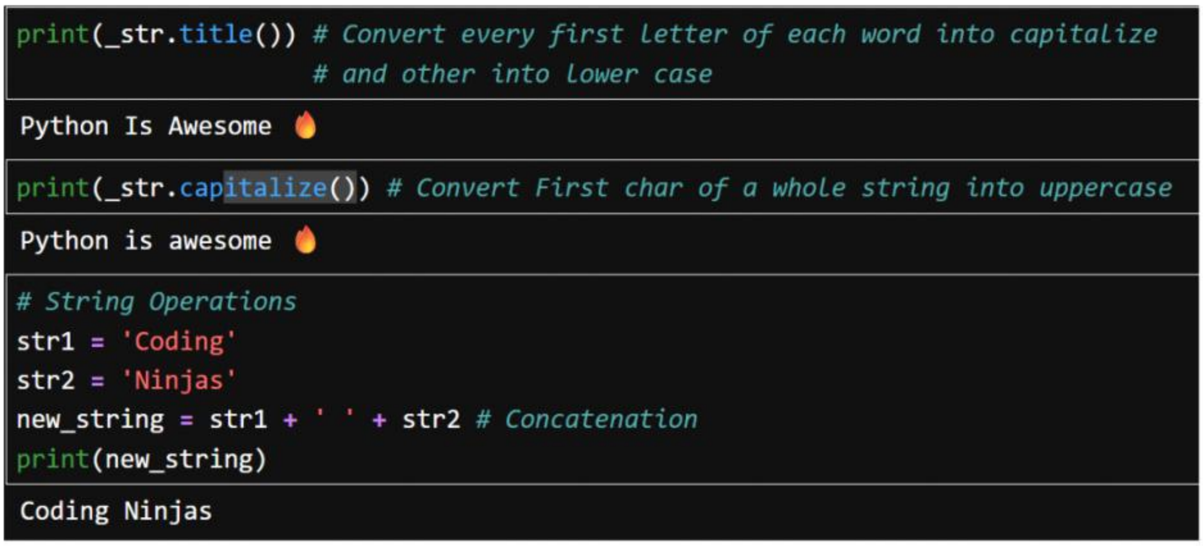
```
PYTHON IS AWESOME 🔥
```

```
print(_str.lower())
```

```
python is awesome 🔥
```

```
_str = _str.lower()
print(_str)
```

```
python is awesome 🔥
```



```python
print(_str.title()) # Convert every first letter of each word into capitalize
                    # and other into lower case
```

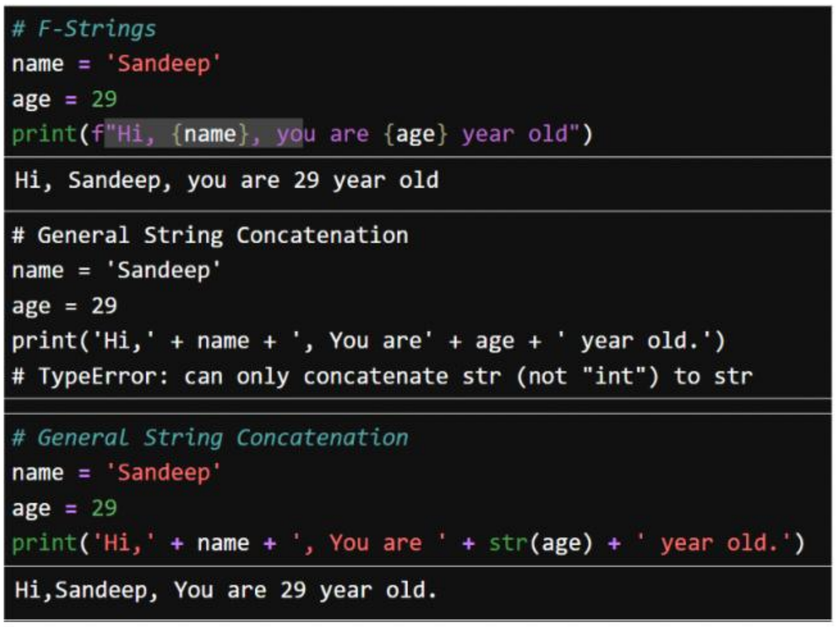
```
Python Is Awesome 🔥
```

```python
print(_str.capitalize()) # Convert First char of a whole string into uppercase
```

```
Python is awesome 🔥
```

```python
# String Operations
str1 = 'Coding'
str2 = 'Ninjas'
new_string = str1 + ' ' + str2 # Concatenation
print(new_string)
```

```
Coding Ninjas
```



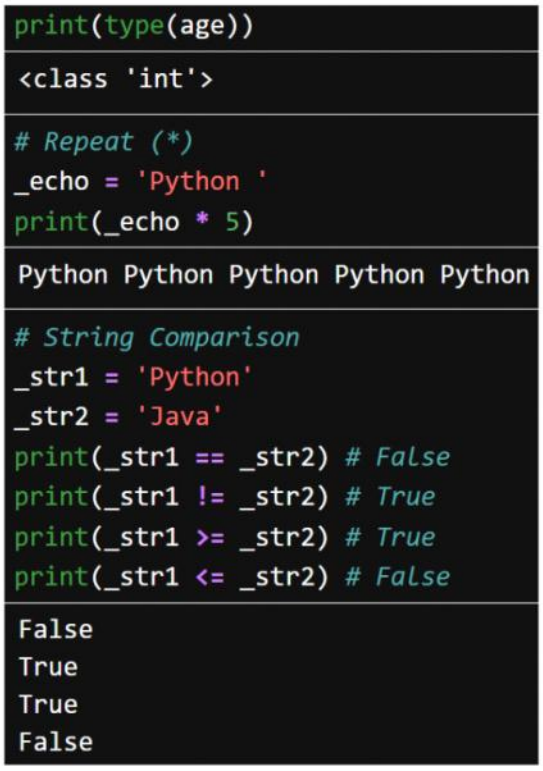
```python
# F-Strings
name = 'Sandeep'
age = 29
print(f"Hi, {name}, you are {age} year old")
```

```
Hi, Sandeep, you are 29 year old
```

```python
# General String Concatenation
name = 'Sandeep'
age = 29
print('Hi,' + name + ', You are' + age + ' year old.')
# TypeError: can only concatenate str (not "int") to str
```

```python
# General String Concatenation
name = 'Sandeep'
age = 29
print('Hi,' + name + ', You are ' + str(age) + ' year old.')
```

```
Hi,Sandeep, You are 29 year old.
```



```python
print(type(age))
```

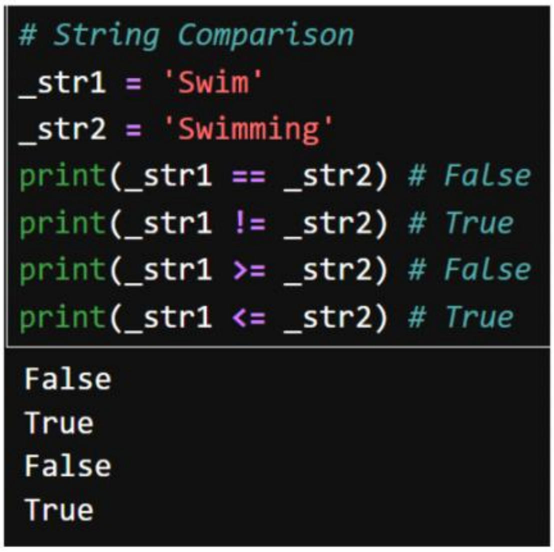
```
<class 'int'>
```

```python
# Repeat (*)
_echo = 'Python '
print(_echo * 5)
```

```
Python Python Python Python Python
```

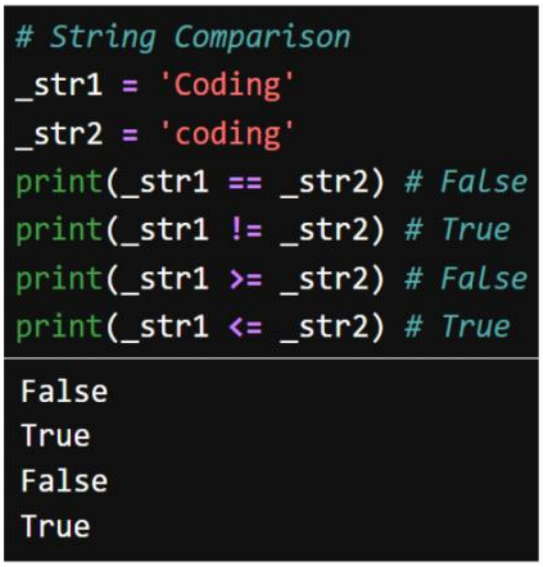
```python
# String Comparison
_str1 = 'Python'
_str2 = 'Java'
print(_str1 == _str2) # False
print(_str1 != _str2) # True
print(_str1 >= _str2) # True
print(_str1 <= _str2) # False
```

```
False
True
True
False
```



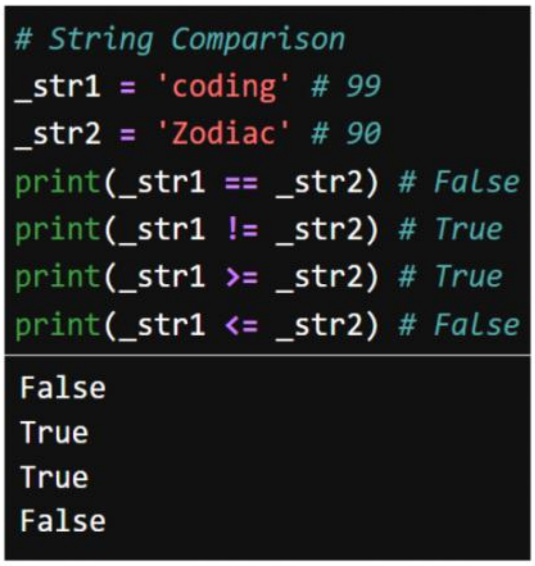
```python
# String Comparison
_str1 = 'Swim'
_str2 = 'Swimming'
print(_str1 == _str2) # False
print(_str1 != _str2) # True
print(_str1 >= _str2) # False
print(_str1 <= _str2) # True
```

```
False
True
False
True
```



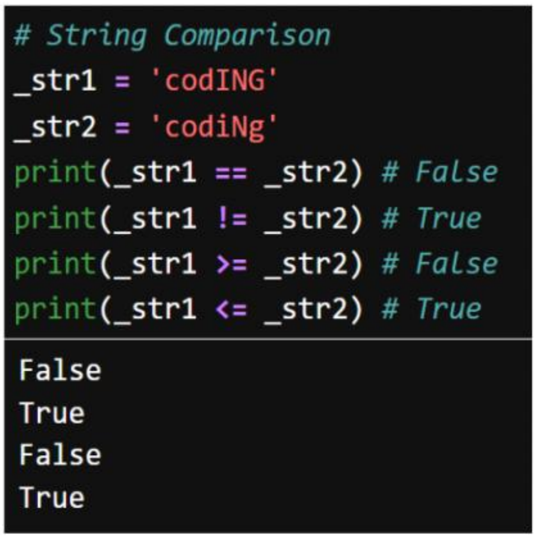
```python
# String Comparison
_str1 = 'Coding'
_str2 = 'coding'
print(_str1 == _str2) # False
print(_str1 != _str2) # True
print(_str1 >= _str2) # False
print(_str1 <= _str2) # True
```

```
False
True
False
True
```



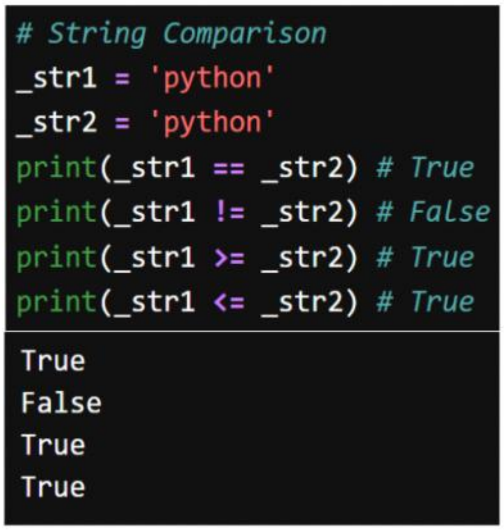
```python
# String Comparison
_str1 = 'coding' # 99
_str2 = 'Zodiac' # 90
print(_str1 == _str2) # False
print(_str1 != _str2) # True
print(_str1 >= _str2) # True
print(_str1 <= _str2) # False
```

```
False
True
True
False
```



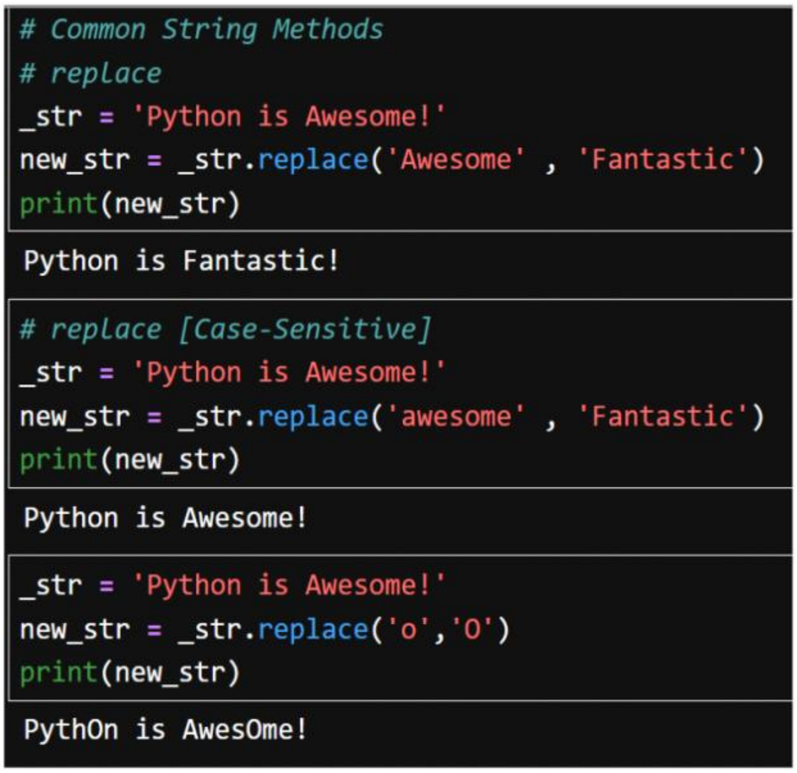
```python
# String Comparison
_str1 = 'codING'
_str2 = 'codiNg'
print(_str1 == _str2) # False
print(_str1 != _str2) # True
print(_str1 >= _str2) # False
print(_str1 <= _str2) # True
```

```
False
True
False
True
```



```python
# String Comparison
_str1 = 'python'
_str2 = 'python'
print(_str1 == _str2) # True
print(_str1 != _str2) # False
print(_str1 >= _str2) # True
print(_str1 <= _str2) # True
```

```
True
False
True
True
```



```python
# Common String Methods
# replace
_str = 'Python is Awesome!'
new_str = _str.replace('Awesome' , 'Fantastic')
print(new_str)
```

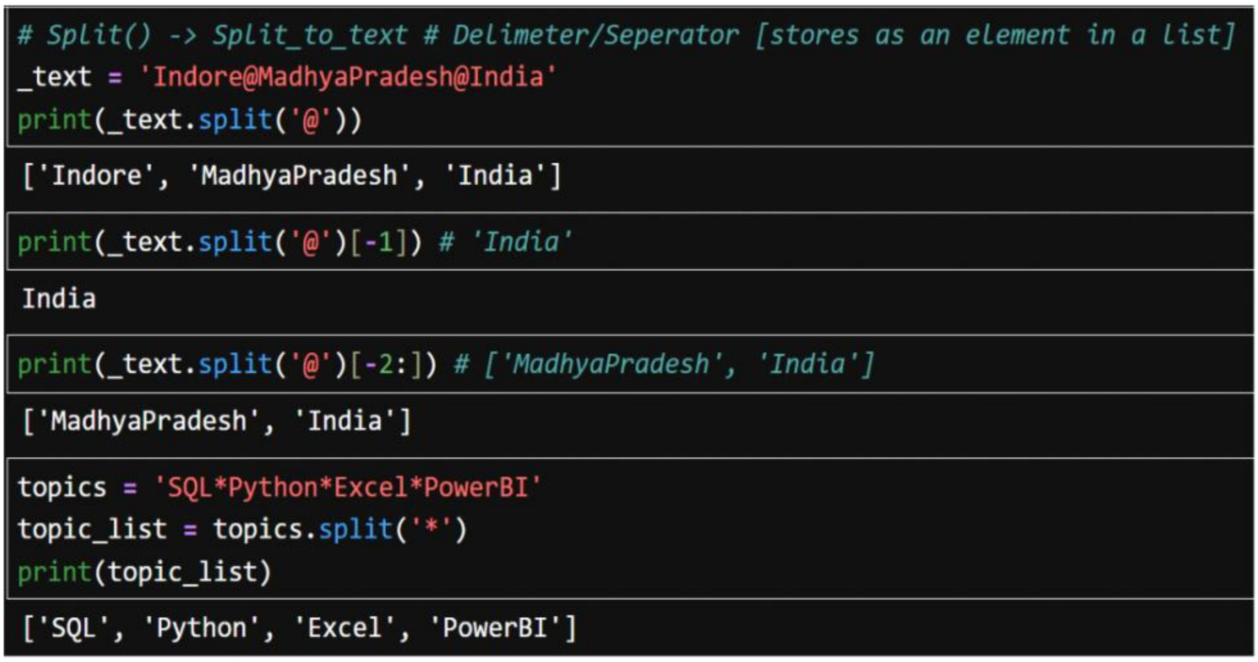
```
Python is Fantastic!
```

```python
# replace [Case-Sensitive]
_str = 'Python is Awesome!'
new_str = _str.replace('awesome' , 'Fantastic')
print(new_str)
```

```
Python is Awesome!
```

```python
_str = 'Python is Awesome!'
new_str = _str.replace('o','O')
print(new_str)
```

```
PythOn is AwesOme!
```



```
# Split() -> Split_to_text # Delimeter/Seperator [stores as an element in a List]
_text = 'Indore@MadhyaPradesh@India'
print(_text.split('@'))
```

```
['Indore', 'MadhyaPradesh', 'India']
```

```
print(_text.split('@')[-1]) # 'India'
```

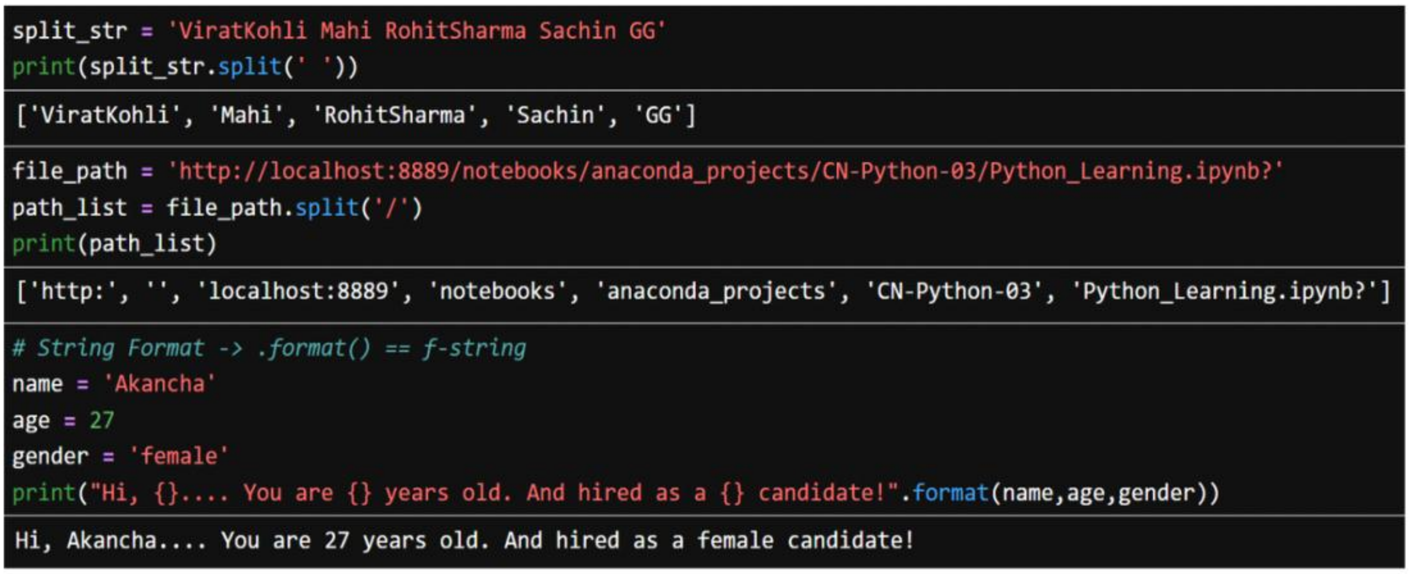
```
India
```

```
print(_text.split('@')[-2:]) # ['MadhyaPradesh', 'India']
```

```
['MadhyaPradesh', 'India']
```

```
topics = 'SQL*Python*Excel*PowerBI'
topic_list = topics.split('*')
print(topic_list)
```

```
['SQL', 'Python', 'Excel', 'PowerBI']
```



```
split_str = 'ViratKohli Mahi RohitSharma Sachin GG'
print(split_str.split(' '))
```

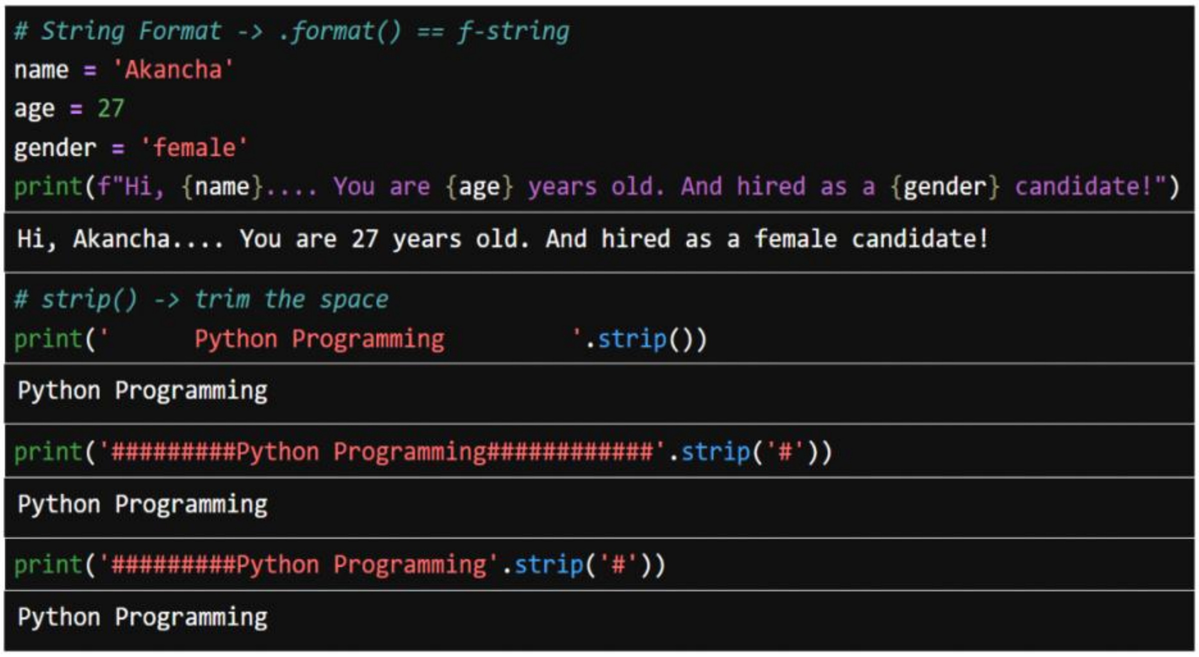
```
['ViratKohli', 'Mahi', 'RohitSharma', 'Sachin', 'GG']
```

```
file_path = 'http://localhost:8889/notebooks/anaconda_projects/CN-Python-03/Python_Learning.ipynb?'
path_list = file_path.split('/')
print(path_list)
```

```
['http:', '', 'localhost:8889', 'notebooks', 'anaconda_projects', 'CN-Python-03', 'Python_Learning.ipynb?']
```

```
# String Format -> .format() == f-string
name = 'Akancha'
age = 27
gender = 'female'
print("Hi, {}.... You are {} years old. And hired as a {} candidate!".format(name,age,gender))
```

```
Hi, Akancha.... You are 27 years old. And hired as a female candidate!
```



```
# String Format -> .format() == f-string
name = 'Akancha'
age = 27
gender = 'female'
print(f"Hi, {name}.... You are {age} years old. And hired as a {gender} candidate!")
```

```
Hi, Akancha.... You are 27 years old. And hired as a female candidate!
```

```
# strip() -> trim the space
print('      Python Programming         '.strip())
```

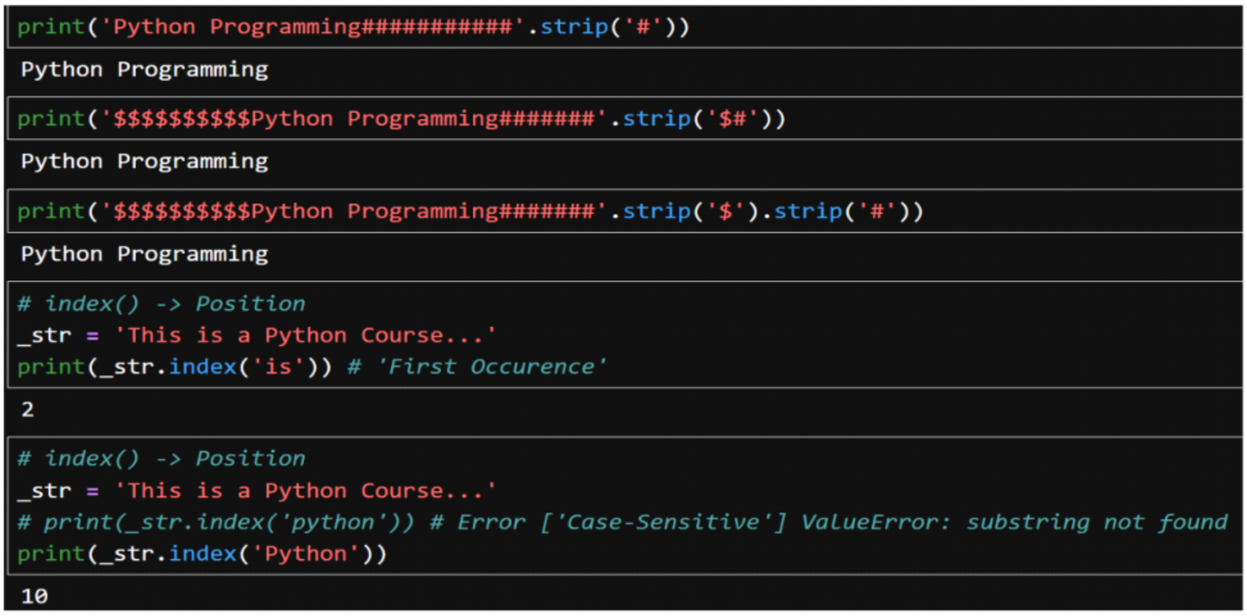
```
Python Programming
```

```
print('#########Python Programming############'.strip('#'))
```

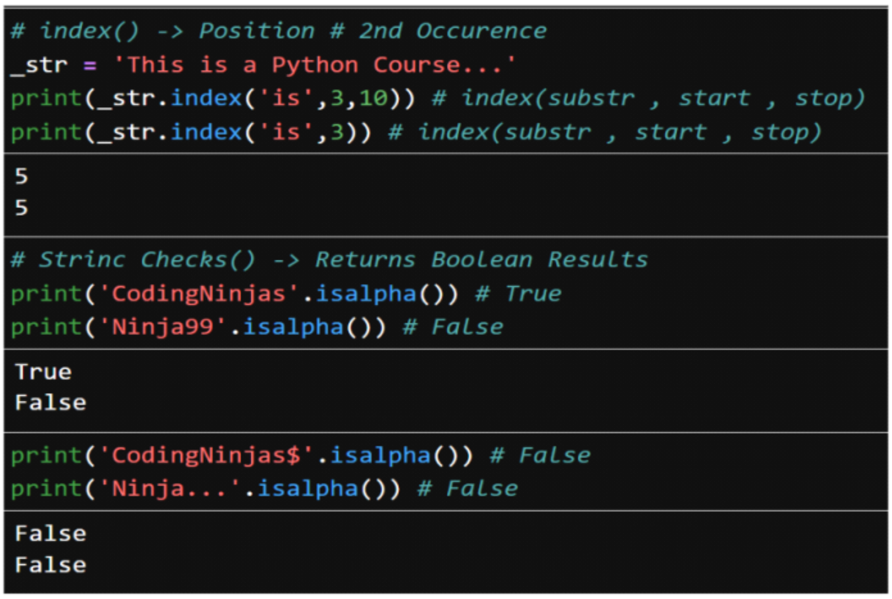
```
Python Programming
```

```
print('#########Python Programming'.strip('#'))
```

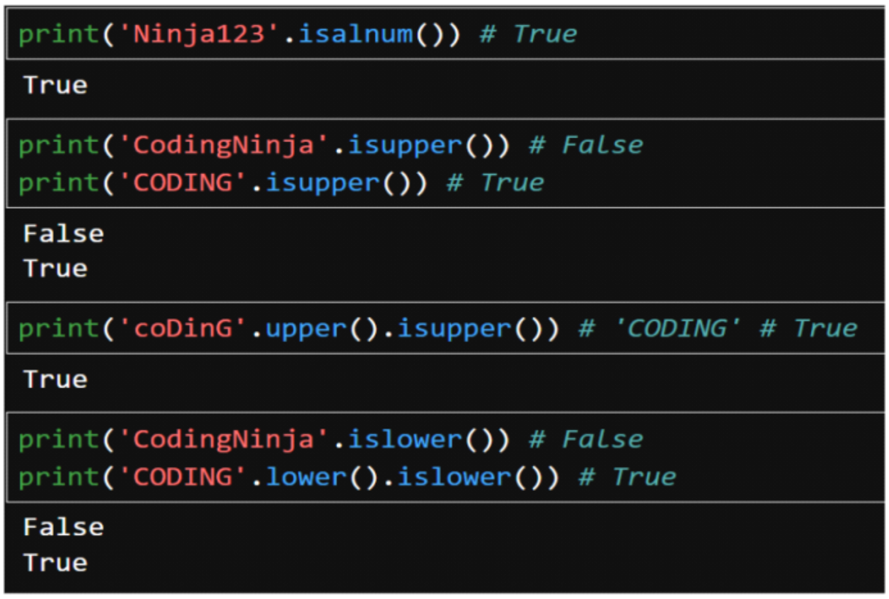
```
Python Programming
```



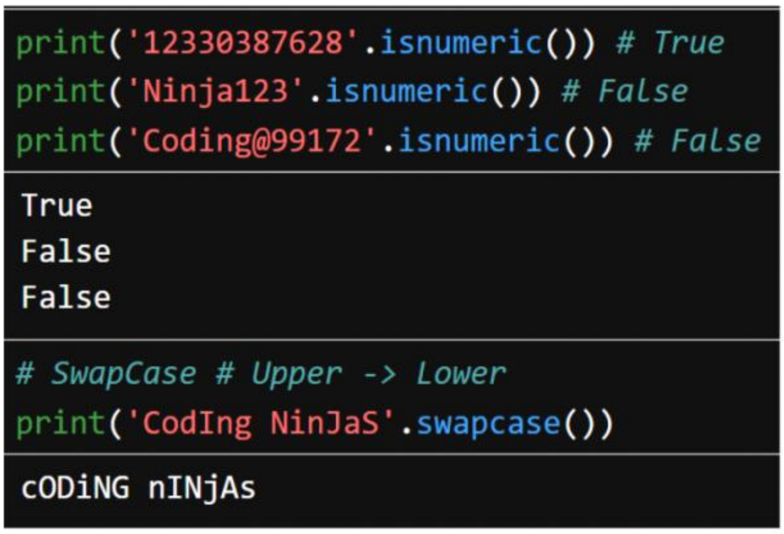
```
print('Python Programming###########'.strip('#'))
```
```
Python Programming
```
```
print('$$$$$$$$$$Python Programming#######'.strip('$#'))
```
```
Python Programming
```
```
print('$$$$$$$$$$Python Programming#######'.strip('$').strip('#'))
```
```
Python Programming
```
```
# index() -> Position
_str = 'This is a Python Course...'
print(_str.index('is')) # 'First Occurence'
```
```
2
```
```
# index() -> Position
_str = 'This is a Python Course...'
# print(_str.index('python')) # Error ['Case-Sensitive'] ValueError: substring not found
print(_str.index('Python'))
```
```
10
```



```
# index() -> Position # 2nd Occurence
_str = 'This is a Python Course...'
print(_str.index('is',3,10)) # index(substr , start , stop)
print(_str.index('is',3)) # index(substr , start , stop)
```
```
5
5
```
```
# Strinc Checks() -> Returns Boolean Results
print('CodingNinjas'.isalpha()) # True
print('Ninja99'.isalpha()) # False
```
```
True
False
```
```
print('CodingNinjas$'.isalpha()) # False
print('Ninja...'.isalpha()) # False
```
```
False
False
```



```
print('Ninja123'.isalnum()) # True
```
```
True
```
```
print('CodingNinja'.isupper()) # False
print('CODING'.isupper()) # True
```
```
False
True
```
```
print('coDinG'.upper().isupper()) # 'CODING' # True
```
```
True
```
```
print('CodingNinja'.islower()) # False
print('CODING'.lower().islower()) # True
```
```
False
True
```

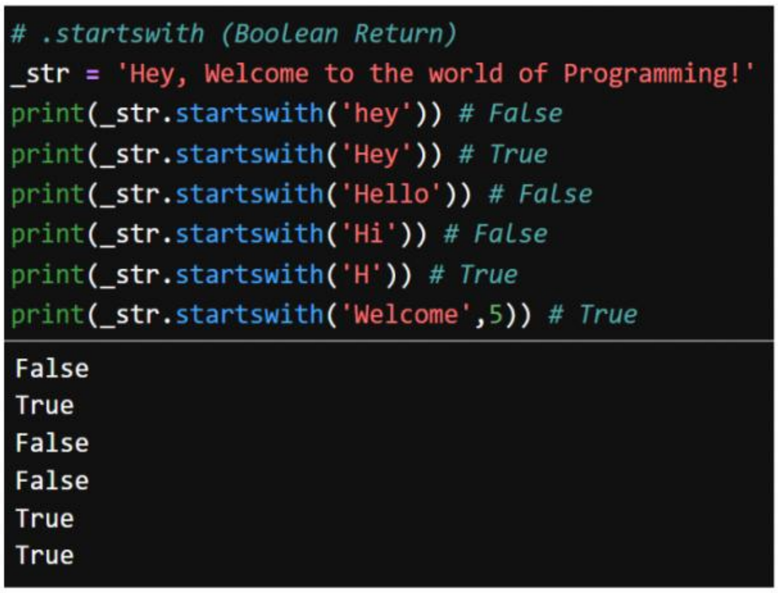


```python
print('12330387628'.isnumeric()) # True
print('Ninja123'.isnumeric()) # False
print('Coding@99172'.isnumeric()) # False
```

```
True
False
False
```

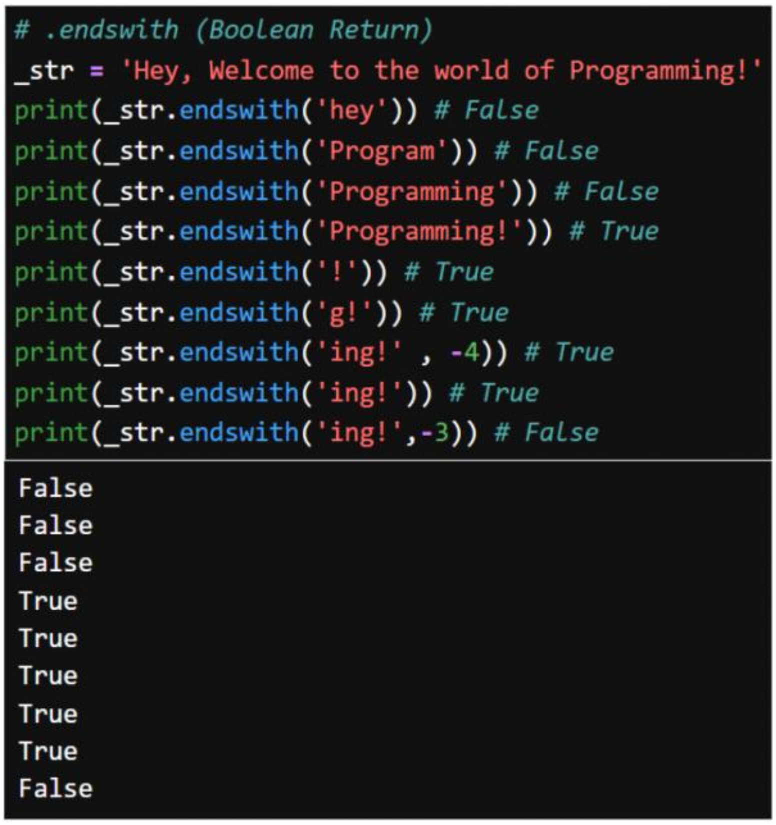
```python
# SwapCase # Upper -> Lower
print('CodIng NinJaS'.swapcase())
```

```
cODiNG nINjAs
```



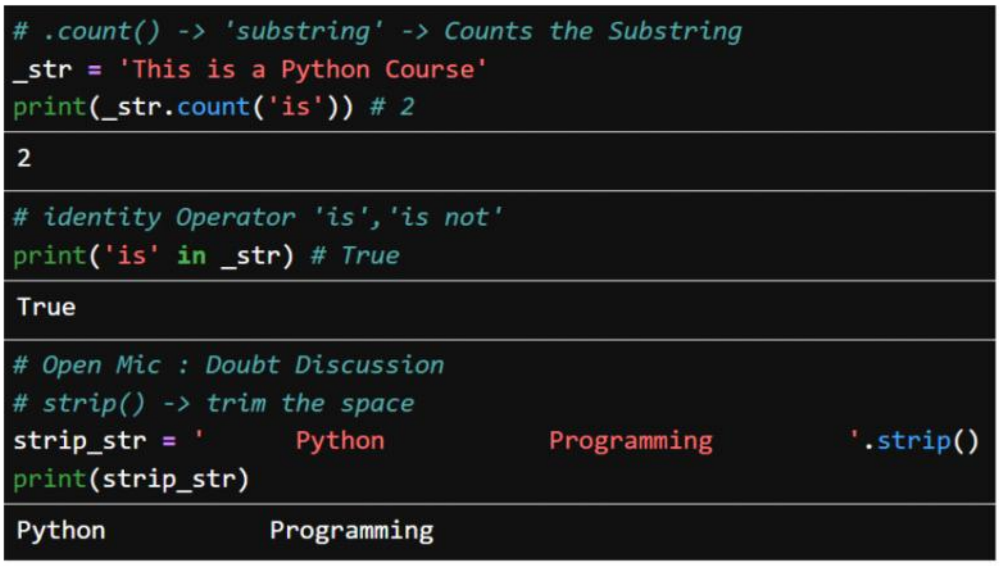
```python
# .startswith (Boolean Return)
_str = 'Hey, Welcome to the world of Programming!'
print(_str.startswith('hey')) # False
print(_str.startswith('Hey')) # True
print(_str.startswith('Hello')) # False
print(_str.startswith('Hi')) # False
print(_str.startswith('H')) # True
print(_str.startswith('Welcome',5)) # True
```

```
False
True
False
False
True
True
```



```python
# .endswith (Boolean Return)
_str = 'Hey, Welcome to the world of Programming!'
print(_str.endswith('hey')) # False
print(_str.endswith('Program')) # False
print(_str.endswith('Programming')) # False
print(_str.endswith('Programming!')) # True
print(_str.endswith('!')) # True
print(_str.endswith('g!')) # True
print(_str.endswith('ing!' , -4)) # True
print(_str.endswith('ing!')) # True
print(_str.endswith('ing!',-3)) # False
```

```
False
False
False
True
True
True
True
True
False
```



```
# .count() -> 'substring' -> Counts the Substring
_str = 'This is a Python Course'
print(_str.count('is')) # 2
```

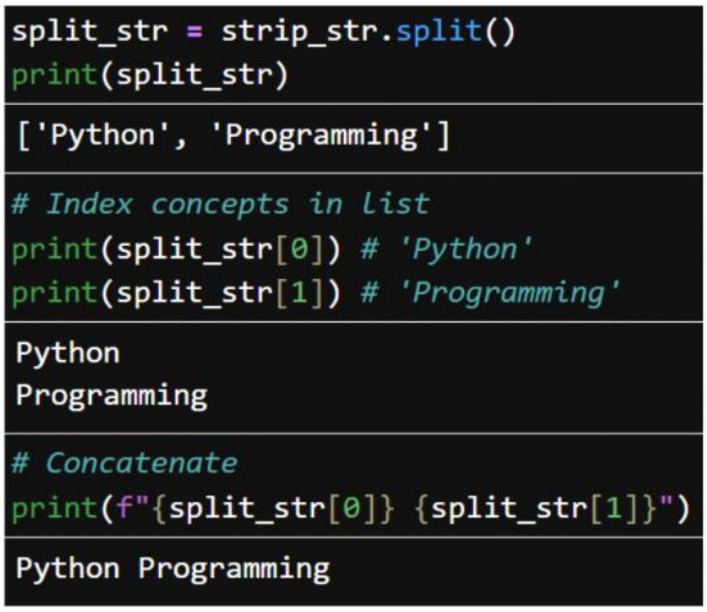
```
2
```

```
# identity Operator 'is','is not'
print('is' in _str) # True
```

```
True
```

```
# Open Mic : Doubt Discussion
# strip() -> trim the space
strip_str = '      Python          Programming          '.strip()
print(strip_str)
```

```
Python          Programming
```



```
split_str = strip_str.split()
print(split_str)
```

```
['Python', 'Programming']
```

```
# Index concepts in list
print(split_str[0]) # 'Python'
print(split_str[1]) # 'Programming'
```

```
Python
Programming
```

```
# Concatenate
print(f"{split_str[0]} {split_str[1]}")
```

```
Python Programming
```