Single line comment

In [1]:
```
letter= 'P'
```

In [2]:
```
print(letter)
```

```
P
```

In [3]:
```
print(len(letter))
```

```
1
```

In [4]:
```
greeting='Hello, World!'
```

In [5]:
```
print(greeting)
```

```
Hello, World!
```

In [6]:
```
print(len(greeting))
```

```
13
```

In [7]:
```
sentence = "I hope you are enjoying 30 days of python challenge"
```

In [8]:
```
print(sentence)
```

```
I hope you are enjoying 30 days of python challenge
```

Multiline String

In [11]:
```
multiline_string = '''I am a teacher and enjoy teaching.
I didn't find anything as rewarding as empowering people.
That is why I created 30 days of python.'''
```

In [10]:
```
print(multiline_string)
```

```
I am a teacher and enjoy teaching.
I didn't find anything as rewarding as empowering people.
That is why I created 30 days of python.
```

Another way of doing the same thing

In [12]:
```
multiline_string = """I am a teacher and enjoy teaching.
I didn't find anything as rewarding as empowering people.
That is why I created 30 days of python."""
```

In [13]:
```
print(multiline_string)
```

```
I am a teacher and enjoy teaching.
I didn't find anything as rewarding as empowering people.
That is why I created 30 days of python.
```

String Concatenation

In [14]:

```
first_name = 'Asabeneh'
```

In [15]:

```
last_name = 'Yetayeh'
space = ' '
full_name = first_name  +  space + last_name
```

In [16]:

```
print(full_name)
```

```
Asabeneh Yetayeh
```

Checking length of a string using len() builtin function

In [17]:

```
print(len(first_name))
```

```
8
```

In [18]:

```
print(len(last_name))
```

```
7
```

In [19]:

```
print(len(first_name) > len(last_name))
```

```
True
```

In [20]:

```
print(len(full_name))
```

```
16
```

Unpacking characters

In [21]:

```
language = 'Python'
```

In [27]:

```
a,b,c,d,e,f= language
```

print(a)

In [28]:

```
print(b)
```

```
y
```

In [29]:

```
print(c)
```

```
t
```

In [30]:

```
print(d)
```

```
h
```

In [31]:

```
print(e)
```

```
o
```

In [32]:

```
print(f)
```

```
n
```

Accessing characters in strings by index

```
In [34]: language = 'Python'
```

```
In [35]: first_letter = language[0]
```

```
In [36]: print(first_letter)
```

```
P
```

```
In [37]: second_letter = language[1]
```

```
In [38]: print(second_letter)
```

```
y
```

```
In [39]: last_index = len(language) - 1
```

```
In [40]: last_letter = language[last_index]
```

```
In [41]: print(last_letter)
```

```
n
```

```
In [42]: print(last_index)
```

```
5
```

```
In [44]: print(language[5])
```

```
n
```

If we want to start from right end we can use negative indexing. -1 is the last index

```
In [45]: language = 'Python'
```

```
In [46]: last_letter = language[-1]
```

```
In [47]: print(last_letter) # n
```

```
n
```

```
In [48]: second_last = language[-2]
```

```
In [49]: print(second_last)
```

```
o
```

```
In [52]: print(language[-6])
```

```
P
```

Slicing

```
In [53]: language = 'Python'
```

```
In [54]: first_three = language[0:3]
```

```
In [55]: last_three = language[3:6]
```

```
In [56]: print(last_three)
```

```
hon
```

```
In [57]: last_three = language[-3:]
```

```
In [58]: print(last_three)
```

```
hon
```

```
In [59]: last_three = language[3:]
```

```
In [60]: print(last_three)
```

```
hon
```

Skipping character while splitting Python strings

```
In [61]: language = 'Python'
```

```
In [62]: pto = language[0:6:2]
```

```
In [63]: print(pto)
```

```
Pto
```

```
In [64]: # Escape sequence
```

```
In [65]: print('I hope every one enjoying the python challenge.\nDo you ?')
```

```
I hope every one enjoying the python challenge.
Do you ?
```

```
In [66]: print('Days\tTopics\tExercises')
```

```
Days    Topics  Exercises
```

```
In [67]: print('Day 1\t3\t5')
```

```
Day 1   3       5
```

```
In [68]: print('Day 2\t3\t5')
```

```
Day 2   3       5
```

```
In [69]: print('Day 3\t3\t5')
```

```
Day 3   3       5
```

```
In [70]: print('Day 4\t3\t5')
```

```
Day 4   3       5
```

```
In [71]: print('This is a back slash  symbol (\\)')
```

```
This is a back slash  symbol (\)
```

```
In [72]: print('In every programming language it starts with \"Hello, World!\"')
```

```
In every programming language it starts with "Hello, World!"
```

```
In [73]: print("In every programming language it starts with 'Hello, World!'")
```

```
In every programming language it starts with 'Hello, World!'
```

String Methods

capitalize(): Converts the first character the string to Capital Letter

```
In [75]: challenge = 'thirty days of python'
```

```
In [79]: print(challenge.capitalize())
```

```
Thirty days of python
```

count(): returns occurrences of substring in string, count(substring, start=.., end=..)

```
In [81]: challenge = 'thirty days of python'
```

```
In [82]: print(challenge.count('y'))
```

```
3
```

```
In [83]: print(challenge.count('y', 7, 14))
```

```
1
```

```
In [84]: print(challenge.count('th'))
```

```
2
```

endswith(): Checks if a string ends with a specified ending

```
In [85]: challenge = 'thirty days of python'
```

```
In [86]: print(challenge.endswith('on'))
```

```
True
```

In [87]:
```python
print(challenge.endswith('thon'))
```
```
True
```

In [88]:
```python
print(challenge.endswith('ron'))
```
```
False
```

In [90]:
```python
print(challenge.endswith('tion'))
```
```
False
```

In [91]:
```python
print(challenge.endswith('t hon'))
```
```
False
```

expandtabs(): Replaces tab character with spaces, default tab size is 8. It takes tab size argument

In [93]:
```python
challenge = 'thirty\tdays\tof\tpython'
```

In [94]:
```python
print(challenge.expandtabs())
```
```
thirty  days    of      python
```

In [95]:
```python
print(challenge.expandtabs(10))
```
```
thirty    days      of        python
```

find(): Returns the index of first occurrence of substring

In [96]:
```python
challenge = 'thirty days of python'
```

In [97]:
```python
print(challenge.find('y'))
```
```
5
```

In [98]:
```python
print(challenge.find('th'))
```
```
0
```

format() formats string into nicer output

In [100…
```python
first_name = 'Siddharth'
last_name = 'Bose'
job = 'Data Scientist'
country = 'India'
sentence = 'I am {} {}. I am a {}. I live in {}.'.format(first_name, last_name,
print(sentence)
```
```
I am Siddharth Bose. I am a Data Scientist. I live in India.
```

In [104…

```
radius = 10
pi = 3.14
area = pi * (radius**2)
result='The area of circle with radius {} is {}'.format(str(radius),str(area))
print(result)
```

```
The area of circle with radius 10 is 314.0
```

index(): Returns the index of substring

In [105…

```
challenge = 'thirty days of python'
print(challenge.find('y'))
print(challenge.find('th'))
```

```
5
0
```

isalnum(): Checks if string contains only alphanumeric character

In [106…

```
challenge = 'ThirtyDaysPython'
print(challenge.isalnum())
```

```
True
```

In [107…

```
challenge = '30DaysPython'
print(challenge.isalnum())
```

```
True
```

In [108…

```
challenge = 'thirty days of python'
print(challenge.isalnum())
```

```
False
```

In [109…

```
challenge = 'thirty days of python 2019'
print(challenge.isalnum())
```

```
False
```

isalpha(): Checks if all characters are alphabets

In [110…

```
challenge = 'thirty days of python'
print(challenge.isalpha())
```

```
False
```

In [111…

```
num = '123'
print(num.isalpha())
```

```
False
```

In [116…

```
grow = 'thirty days of python 2019'
print(grow.isalpha())
```

```
False
```

isdecimal(): Checks Decimal Characters

In [126…

```
challenge = "thirty days of python 2019"
print(challenge.isdecimal())
```

```
False
```

isdigit(): Checks Digit Characters

In [127…

```
challenge = 'Thirty'
print(challenge.isdigit())
```

```
False
```

In [129…

```
challenge = '30'
print(challenge.isdigit())
```

```
True
```

isdecimal():Checks decimal characters

In [1]:

```
num = '10'
print(num.isdecimal())
```

```
True
```

In [3]:

```
num='10.5'
print(num.isdecimal())
```

```
False
```

isidentifier():Checks for valid identifier means it check if a string is a valid variable name

In [4]:

```
challenge = '30DaysOfPython'
print(challenge.isidentifier())
```

```
False
```

In [5]:

```
challenge = 'thirty_days_of_python'
print(challenge.isidentifier())
```

```
True
```

islower():Checks if all alphabets in a string are lowercase

In [6]:

```
challenge = 'thirty days of python'
print(challenge.islower())
```

```
True
```

In [7]:

```
challenge = 'Thirty days of python'
print(challenge.islower())
```

```
False
```

isupper(): returns if all characters are uppercase characters

In [8]:

```
challenge = 'thirty days of python'
print(challenge.isupper())
```

```
False
```

In [9]:

```
challenge = 'THIRTY DAYS OF PYTHON'
print(challenge.isupper())
```

```
True
```

isnumeric():Checks numeric characters

In [11]:

```
num = '10'
print(num.isnumeric())
```

```
True
```

In [12]:

```
print('ten'.isnumeric())
```

```
False
```

join(): Returns a concatenated string

In [16]:

```
web_tech = ['HTML', 'CSS', 'JavaScript', 'React']
result = '#, '.join(web_tech)
```

In [15]:

```
print(result)
```

```
HTML#, CSS#, JavaScript#, React
```

strip(): Removes both leading and trailing characters

In [19]:

```
challenge = '   thirty days of python  '
print(challenge.strip('y'))
```

```
   thirty days of python
```

replace(): Replaces substring inside

In [20]:

```
challenge = 'thirty days of python'
print(challenge.replace('python', 'coding'))
```

```
thirty days of coding
```

split():Splits String from Left

In [22]:

```
challenge = 'thirty days of python'
print(challenge.split()) # ['thirty', 'days', 'of', 'python']
```

```
['thirty', 'days', 'of', 'python']
```

In [23]:

```
challenge = '300 days mght20 of python'
print(challenge.split())
```

```
['300', 'days', 'mght20', 'of', 'python']
```

title(): Returns a Title Cased String

In [25]:

```
challenge = 'thirty days of python'
print(challenge.title())
```

```
Thirty Days Of Python
```

swapcase(): Checks if String Starts with the Specified String

In [27]:

```
challenge = 'thirty days of python'
print(challenge.swapcase())
```

```
THIRTY DAYS OF PYTHON
```

In [28]:

```
challenge = 'Thirty Days Of Python'
print(challenge.swapcase())
```

```
tHIRTY dAYS oF pYTHON
```

startswith(): Checks if String Starts with the Specified String

In [29]:

```
challenge = 'thirty days of python'
print(challenge.startswith('thirty'))
```

```
True
```

In [30]:

```
challenge = '30 days of python'
print(challenge.startswith('thirty'))
```

```
False
```