In [1]:
```
letter = 'P'                # A string could be a single character or a bunch
```

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

```
P
```

In [3]:
```
print(len(letter))          # 1
```

```
1
```

In [4]:
```
greeting = 'Hello, World!'  # String could be  a single or double quote,"Hello
```

In [5]:
```
print(greeting)             # Hello, World!
```

```
Hello, World!
```

In [6]:
```
print(len(greeting))        # 13
```

```
13
```

In [8]:
```
sentence = "I am enjoying 30 days of python challenge"
```

In [9]:
```
print(sentence)
```

```
I am enjoying 30 days of python challenge
```

Multiline String

In [10]:
```
multiline_string = '''I am a student and enjoy learning.
I didn't find anything as rewarding as empowering people.
That is why I created 30 days of python.'''
```

In [11]:
```
print(multiline_string)
```

```
I am a student and enjoy learning.
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 student and enjoy learning.
I didn't find anything as rewarding as empowering people.
That is why I created 30 days of python."""
print(multiline_string)
```

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

String Concatenation

In [13]:
```
first_name = 'Rahul'
last_name = 'Vishwakarma'
space = ' '
full_name = first_name  +  space + last_name
```

In [14]:
```
print(full_name) # Rahul vishwakarma
```

```
Rahul Vishwakarma
```

Checking length of a string using len() builtin function

In [15]:
```
print(len(first_name))  # 5
```

```
5
```

In [17]:
```
print(len(last_name))   # 11
```

```
11
```

In [18]:
```
print(len(first_name) > len(last_name)) # False
```

```
False
```

In [20]:
```
print(len(full_name)) # 17
```

```
17
```

Unpacking characters

In [21]:
```
language = 'Python'
```

In [22]:
```
a,b,c,d,e,f = language # unpacking sequence characters into variables
```

In [23]:
```
print(a) # P
```

```
P
```

In [24]:
```
print(b) # y
```

```
y
```

In [25]:
```
print(c) # t
```

```
t
```

In [26]:
```
print(d) # h
```

```
h
```

In [27]:
```
print(e) # o
```

```
o
```

In [28]:
```
print(f) # n
```

```
n
```

Accessing characters in strings by index

In [29]:
```
language = 'Python'
```

In [30]:
```
first_letter = language[0]
```

In [31]:
```
print(first_letter) # P
```

```
P
```

In [32]:
```
second_letter = language[1]
```

In [33]:
```
print(second_letter) # y
```

```
y
```

In [34]:
```
last_index = len(language) - 1
```

In [35]:
```
last_letter = language[last_index]
```

In [36]:
```
print(last_letter) # n
```

```
n
```

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

In [37]:
```
language = 'Python'
```

In [38]:
```
last_letter = language[-1]
```

In [39]:
```
print(last_letter) # n
```

```
n
```

In [40]:
```
second_last = language[-2]
```

In [41]:
```
print(second_last) # o
```

```
o
```

Slicing

In [42]:
```
language = 'Python'
first_three = language[0:3] # starts at zero index and up to 3 but not include
last_three = language[3:6]
print(last_three) # hon
```

```
hon
```

Another way

In [44]:
```
last_three = language[-3:]
print(last_three)   # hon
```

```
hon
```

In [45]:
```
last_three = language[3:]
print(last_three)   # hon
```

```
hon
```

Skipping character while splitting Python strings

In [46]:
```
language = 'Python'
```

In [47]:
```
pto = language[0:6:2] #
```

In [48]:
```
print(pto) # pto
```

```
Pto
```

Escape sequence

In [49]:
```
print('I hope every one enjoying the python challenge.\nDo you ?') # line brea
```

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

In [50]:
```
print('Days\tTopics\tExercises')
```

```
Days        Topics        Exercises
```

In [51]:
```
print('Day 1\t3\t5')
```

```
Day 1        3        5
```

In [52]:
```
print('Day 2\t3\t5')
```

```
Day 2	3	5
```

In [53]:
```
print('Day 3\t3\t5')
```

```
Day 3	3	5
```

In [54]:
```
print('Day 4\t3\t5')
```

```
Day 4	3	5
```

In [55]:
```
print('This is a back slash  symbol (\\)') # To write a back slash
```

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

In [56]:
```
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 [57]:
```
challenge = 'thirty days of python'
```

In [58]:
```
print(challenge.capitalize()) # 'Thirty days of python'
```

```
Thirty days of python
```

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

In [59]:
```
challenge = 'thirty days of python'
```

In [60]:
```
print(challenge.count('y')) # 3
```

```
3
```

In [61]:
```
print(challenge.count('y', 7, 14)) # 1
```

```
1
```

In [62]:
```
print(challenge.count('th')) # 2`
```

```
2
```

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

In [63]:
```
challenge = 'thirty days of python'
```

In [64]:
```
print(challenge.endswith('on'))   # True
```

```
True
```

In [65]:
```
print(challenge.endswith('tion')) # False
```

```
False
```

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

In [66]:
```
challenge = 'thirty\tdays\tof\tpython'
```

In [67]:
```
print(challenge.expandtabs())   # 'thirty  days    of      python'
```

```
thirty  days    of      python
```

In [68]:
```
print(challenge.expandtabs(10)) # 'thirty    days      of        python'
```

```
thirty    days      of        python
```

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

In [69]:
```
challenge = 'thirty days of python'
```

In [70]:
```
print(challenge.find('y'))  # 5
```

```
5
```

In [71]:
```
print(challenge.find('th')) # 0
```

```
0
```

format() formats string into nicer output

In [74]:
```
first_name = 'Rahul'
```

In [75]:
```
last_name = 'Vishwakarma'
```

In [76]:
```
job = 'teacher'
```

In [77]:
```
country = 'India'
```

In [78]:
```
sentence = 'I am {} {}. I am a {}. I live in {}.'.format(first_name, last_name
```

In [79]:
```
print(sentence) # I am Asabeneh Yetayeh. I am a teacher. I live in Finland.
```

```
I am Rahul Vishwakarma. I am a teacher. I live in India.
```

In [80]:
```
radius = 10
```

In [81]:
```
pi = 3.14
```

In [82]:
```
area = pi # radius ## 2
```

In [83]:
```
result = 'The area of circle with {} is {}'.format(str(radius), str(area))
```

In [84]:
```
print(result) # The area of circle with 10 is 314.0
```

```
The area of circle with 10 is 3.14
```

index(): Returns the index of substring

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

In [86]:
```
print(challenge.find('y'))  # 5
```

```
5
```

In [87]:
```
print(challenge.find('th')) # 0
```

```
0
```

isalnum(): Checks alphanumeric character

In [88]:
```
challenge = 'ThirtyDaysPython'
```

In [89]:
```
print(challenge.isalnum()) # True
```

```
True
```

In [90]:
```
challenge = '30DaysPython'
print(challenge.isalnum()) # True
```

```
True
```

In [91]:
```
challenge = 'thirty days of python'
print(challenge.isalnum()) # False
```

```
False
```

In [92]:
```
challenge = 'thirty days of python 2019'
print(challenge.isalnum()) # False
```

```
False
```

isalpha(): Checks if all characters are alphabets

In [93]:
```
challenge = 'thirty days of python'
print(challenge.isalpha()) # True
num = '123'
print(num.isalpha())      # False
```

```
False
False
```

isdecimal(): Checks Decimal Characters

In [94]:
```
challenge = 'thirty days of python'
print(challenge.find('y'))  # 5
print(challenge.find('th')) # 0
```

```
5
0
```

isdigit(): Checks Digit Characters

In [96]:
```
challenge = 'Thirty'
print(challenge.isdigit()) # False
challenge = '30'
print(challenge.digit())   # True
```

```
False
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[96], line 4
      2 print(challenge.isdigit()) # False
      3 challenge = '30'
----> 4 print(challenge.digit())

AttributeError: 'str' object has no attribute 'digit'
```

In [ ]:
```

```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: