Dealing with Module: Math Module

In [1]:
```
# importing required modules
import math
```

In [11]:
```
print (math.sin(0), math.cos(0), math.tan(0))
print (math.sinh(0), math.cosh(0), math.tanh(0))
print (math.asinh(0.5))
print (math.acosh(1))
print (math.atanh(0.5))
```

```
0.0 1.0 0.0
0.0 1.0 0.0
0.48121182505960347
0.0
0.5493061443340549
```

In [12]:
```
# some constants in the math module
print (math.pi)
print (math.e)
print (math.tau)
```

```
3.141592653589793
2.718281828459045
6.283185307179586
```

In [19]:
```
print (math.log(10), math.log(10, math.e))
print (math.log2(1024), math.log(1024, 2))
print (math.log10(1000), math.log(1000, 10))
```

```
2.302585092994046 2.302585092994046
10.0 10.0
3.0 2.9999999999999996
```

In [22]:
```
print (math.factorial(5), math.factorial(7))
print (math.gcd(100, 750), math.gcd(350, 850))
print (math.pow(10, 3), 10 ** 3, 10.0 ** 3, 10 ** 3.0)
```

```
120 5040
50 50
1000.0 1000 1000.0 1000.0
```

In [24]:
```
print (math.ceil(10.1), math.ceil(10.9), math.floor(10.1), math.floor(10.9))
```

```
11 11 10 10
```

In [26]:
```
help (math.pow)
```

```
Help on built-in function pow in module math:

pow(x, y, /)
    Return x**y (x to the power of y).
```

In [ ]:
```
help (math)
```

Dealing with String Processing

String is a collection of alpha numeric and special characters. String is immutable as insert, delete and update operations can not be carried out on a given string.

In [ ]:
```
# string indexing and slicing
index from left to right ->   0   1   2   3   4   5   6   7   8   9
                   mystr ->   u   n   i   v   e   r   s   i   t   y
index from right to left -> -10  -9  -8  -7  -6  -5  -4  -3  -2  -1
```

In [117…
```
mystr = "university"
print (mystr, len(mystr), type(mystr), id(mystr))
print (mystr[6], mystr[-4], mystr[8], mystr[-2], mystr[9], mystr[-1])  # indexing
print (mystr[1:6], mystr[-9:-4], mystr[1:-4], mystr[-9:6])   # slicing
print (mystr[0:6], mystr[:6], mystr[-10:-4], mystr[:-4])
print (mystr[6:], mystr[-4:])
print (mystr[3:6], mystr[-7:-4], mystr[3:-4], mystr[-7:6])
print (mystr[0::2], mystr[1::2], mystr[::-1])
```

```
university 10 <class 'str'> 2716497143280
s s t t y y
niver niver niver niver
univer univer univer univer
sity sity
ver ver ver ver
uiest nvriy ytisrevinu
```

In [83]:

```
mystr = "sTAndForD unIVerSitY"
print (mystr, len(mystr), type(mystr), id(mystr))
print (mystr.upper())
print (mystr.lower())
print (mystr.swapcase())
print (mystr.capitalize())
print (mystr.title())
print (mystr.center(40))
```

```
sTAndForD unIVerSitY 20 <class 'str'> 2716502973376
STANDFORD UNIVERSITY
standford university
StaNDfORd UNivERsITy
Standford university
Standford University
          sTAndForD unIVerSitY
```

In [48]:

```
mystr = "abcd"
print (mystr, mystr.isalpha(), mystr.isalnum(), mystr.isdigit())
mystr = "1234"
print (mystr, mystr.isalpha(), mystr.isalnum(), mystr.isdigit())
mystr = "1234abcd"
print (mystr, mystr.isalpha(), mystr.isalnum(), mystr.isdigit())
mystr = "abcd@1234"
print (mystr, mystr.isalpha(), mystr.isalnum(), mystr.isdigit())
```

```
abcd True True False
1234 False True True
1234abcd False True False
abcd@1234 False False False
```

In [63]:

```
mystr = "charity begins at home"
print (mystr, len(mystr), type(mystr), id(mystr))
print (mystr.startswith("charity"))
print (mystr.startswith("begi"), mystr.startswith("begi", 8), mystr.startswith("begi", 8, 20), mystr.startswith("beg
print (mystr.endswith("home"), mystr.endswith("at"), mystr.endswith("at", 0, 17), mystr.endswith("at", 15, 17))
```

```
charity begins at home 22 <class 'str'> 2716497272224
True
False True True False
True False True True
```

In [67]:

```
mystr = "charity begins at home"
print (mystr, len(mystr), type(mystr), id(mystr))
print (mystr.find("begins"))
print (mystr.find("at"))
print (mystr.find("school"))
```

```
charity begins at home 22 <class 'str'> 2716497269344
8
15
-1
```

In [70]:

```
mystr = "charity begins at home"
print (mystr, len(mystr), type(mystr), id(mystr))
print (mystr.index("begins"))
print (mystr.index("at"))
print (mystr.index("school"))
```

```
charity begins at home 22 <class 'str'> 2716497436144
8
15
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-70-feaf6260cd2a> in <module>
      3 print (mystr.index("begins"))
      4 print (mystr.index("at"))
----> 5 print (mystr.index("school"))

ValueError: substring not found
```

In [71]:

```
try:
    mystr = "charity begins at home"
    print (mystr, len(mystr), type(mystr), id(mystr))
    print (mystr.index("begins"))
    print (mystr.index("at"))
    print (mystr.index("school"))
except ValueError as ve:
    print ("Unsuccessful searching has taken place...!!!")
    print ("Error message:", ve)
```

```
charity begins at home 22 <class 'str'> 2716497269584
8
15
Unsuccessful searching has taken place...!!!
Error message: substring not found
```

In [73]:

```
mystr = "mississippi"
print (mystr, len(mystr), type(mystr), id(mystr))
print (mystr.count("i"), mystr.count("p"))
```

```
mississippi 11 <class 'str'> 2716503020912
4 2
```

In [76]:

```
mystr = "good morning"
print (mystr, len(mystr), type(mystr), id(mystr))
mystr = mystr.replace("morning", "night")
print (mystr, len(mystr), type(mystr), id(mystr))
```

```
good morning 12 <class 'str'> 2716498502832
good night 10 <class 'str'> 2716498502768
```

In [81]:

```
mystr = "    good    morning   "
print (mystr, len(mystr), type(mystr), id(mystr))
result = mystr.strip()
print (result, len(result))
result = mystr.lstrip()
print (result, len(result))
result = mystr.rstrip()
print (result, len(result))
```

```
    good    morning    22 <class 'str'> 2716497271024
good    morning 15
good    morning    18
    good    morning 19
```

In [82]:

```
mystr = "#@@#good#@##morning@#@"
print (mystr, len(mystr), type(mystr), id(mystr))
result = mystr.strip("#@")
print (result, len(result))
result = mystr.lstrip("@#")
print (result, len(result))
result = mystr.rstrip("#@")
print (result, len(result))
```

```
#@@#good#@##morning@#@ 22 <class 'str'> 2716494737696
good#@##morning 15
good#@##morning@#@ 18
#@@#good#@##morning 19
```

In [86]:

```
mystr = "charity begins at home"
print (mystr, len(mystr), type(mystr), id(mystr))
result = mystr.split()
print (result, type(result))
result = mystr.split("i")
print (result, type(result))
result = mystr.split("x")
print (result, type(result))
```

```
charity begins at home 22 <class 'str'> 2716492723904
['charity', 'begins', 'at', 'home'] <class 'list'>
['char', 'ty beg', 'ns at home'] <class 'list'>
['charity begins at home'] <class 'list'>
```

In [90]:

```
list1 = ['charity', 'begins', 'at', 'home']
print (list1, len(list1), type(list1), id(list1))
mystr = " ".join(list1)
print (mystr, len(mystr), type(mystr), id(mystr))
mystr = ", ".join(list1)
print (mystr, len(mystr), type(mystr), id(mystr))
mystr = " - ".join(list1)
print (mystr, len(mystr), type(mystr), id(mystr))
```

```
['charity', 'begins', 'at', 'home'] 4 <class 'list'> 2716497968640
charity begins at home 22 <class 'str'> 2716487640448
charity, begins, at, home 25 <class 'str'> 2716499342160
charity - begins - at - home 28 <class 'str'> 2716499341920
```

In [97]:

```
mystr = "university"
for i in range(len(mystr)):
    print (mystr[i], end = ", ")

print()

for ch in mystr:
    print (ch, end = ", ")

print ()
```

```
u, n, i, v, e, r, s, i, t, y,
u, n, i, v, e, r, s, i, t, y,
```

In [98]:

```
mystr[3] = "y"
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-98-c4fba89c710f> in <module>
----> 1 mystr[3] = "y"

TypeError: 'str' object does not support item assignment
```

CLASS ASSIGNMENT-2 / Day-2

> Take a sentence with upper and lower case letters and other characters from the user and find the number of vowels and consonants in the given string.

In [157…

```
mystr = input("Please enter a sentence: ").lower()
vcount = ccount = 0
for ch in mystr:
    if (ch.isalpha()):
        if (ch in "aeiou"): vcount += 1
        else: ccount += 1
print (f"So number of vowels is {vcount} and consonant is {ccount}...")
print ("End of the program...")
```

```
So number of vowels is 4 and consonant is 5...
End of the program...
```

In [ ]:

Dealing with the List Processing

> List is a collection of data items of same or different datatypes enclosed with in square brackets. List items are mutable as INSERT, DELETE and UPDATE operations can be carried out on them.

In [101…

```
list1 = [100, 500, 200, 400, 300]
print (list1, len(list1), type(list1), id(list1))
print (max(list1), min(list1))
print (sum(list1), sum(list1) / len(list1))
```

```
[100, 500, 200, 400, 300] 5 <class 'list'> 2716500672192
500 100
1500 300.0
```

In [105…

```
list1 = ["Monday", "Friday", "Thursday", "Tuesday", "Sunday"]
print (list1, len(list1), type(list1), id(list1))
print (max(list1), min(list1))
print (sum(list1), sum(list1) / len(list1))
```

```
['Monday', 'Friday', 'Thursday', 'Tuesday', 'Sunday'] 5 <class 'list'> 2716497400128
Tuesday Friday
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-105-f46b71805974> in <module>
      2 print (list1, len(list1), type(list1), id(list1))
      3 print (max(list1), min(list1))
----> 4 print (sum(list1), sum(list1) / len(list1))

TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

In [106…

```
list1 = ["Monday", 900, "Friday", "Thursday", 700, "Tuesday", "Sunday"]
print (list1, len(list1), type(list1), id(list1))
print (max(list1), min(list1))
print (sum(list1), sum(list1) / len(list1))
```

```
['Monday', 900, 'Friday', 'Thursday', 700, 'Tuesday', 'Sunday'] 7 <class 'list'> 2716498678336
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-106-4461b050e2fb> in <module>
      1 list1 = ["Monday", 900, "Friday", "Thursday", 700, "Tuesday", "Sunday"]
      2 print (list1, len(list1), type(list1), id(list1))
----> 3 print (max(list1), min(list1))
      4 print (sum(list1), sum(list1) / len(list1))

TypeError: '>' not supported between instances of 'int' and 'str'
```

In [109…

```
list1 = [100, 500.45, 200, False, 400.5, True, 300]
print (list1, len(list1), type(list1), id(list1))
print (max(list1), min(list1))
print (sum(list1), sum(list1) / len(list1))
```

```
[100, 500.45, 200, False, 400.5, True, 300] 7 <class 'list'> 2716499104448
500.45 False
1501.95 214.56428571428572
```

In [ ]:

```
# list indexing and slicing
index from left to right ->   0        1          2         3           4
                   list1 -> ["Sunday" "Tuesday", "Friday", "Saturday", "Thursday"]
index from right to left ->  -5       -4         -3        -2          -1
```

In [124…

```
list1 = ["Sunday", "Tuesday", "Friday", "Saturday", "Thursday"]
print (list1[2], list1[-3], list1[4], list1[-1])
print (list1[0::2], list1[1::2], list1[::-1])
print (list1[3][2:5], list1[-2][-6:-3])
```

```
Friday Friday Thursday Thursday
['Sunday', 'Friday', 'Thursday'] ['Tuesday', 'Saturday'] ['Thursday', 'Saturday', 'Friday', 'Tuesday', 'Sunday']
tur tur
```

In [138…

```
# defining empty list, empty tuple, empty dictionary, empty set and empty frozen-set
var1 = []     # empty list
print (var1, len(var1), type(var1), id(var1))
var1 = ()     # empty tuple
print (var1, len(var1), type(var1), id(var1))
var1 = {}     # empty dictionary
print (var1, len(var1), type(var1), id(var1))
var1 = set() # empty set
print (var1, len(var1), type(var1), id(var1))
var1 = frozenset([])   # empty frozen set
print (var1, len(var1), type(var1), id(var1))
var1 = (9,)      # singleton representation of a tuple
print (var1, type(var1), id(var1))
```

```
[] 0 <class 'list'> 2716500198528
() 0 <class 'tuple'> 2716400418880
{} 0 <class 'dict'> 2716498525376
set() 0 <class 'set'> 2716500447008
frozenset() 0 <class 'frozenset'> 2716457736896
(9,) <class 'tuple'> 2716497721952
```

In [144…

```
# converting string to list, tuple, set and frozen set
mystr = "my word mississippi"
print (mystr, len(mystr))
result = list(mystr)
print (result, len(result), type(result), id(result))
result = tuple(mystr)
print (result, len(result), type(result), id(result))
result = set(mystr)
print (result, len(result), type(result), id(result))
result = frozenset(mystr)
print (result, len(result), type(result), id(result))
```

```
my word mississippi 19
['m', 'y', ' ', 'w', 'o', 'r', 'd', ' ', 'm', 'i', 's', 's', 'i', 's', 's', 'i', 'p', 'p', 'i'] 19 <class 'list'> 27
16497758400
('m', 'y', ' ', 'w', 'o', 'r', 'd', ' ', 'm', 'i', 's', 's', 'i', 's', 's', 'i', 'p', 'p', 'i') 19 <class 'tuple'> 2
716497859200
{'m', 'p', 'o', 'i', 'w', ' ', 'd', 'r', 's', 'y'} 10 <class 'set'> 2716504046720
frozenset({'m', 'p', 'o', 'i', 'w', ' ', 'd', 'r', 's', 'y'}) 10 <class 'frozenset'> 2716495411008
```

In [147…

```
# insert opeation on list
list1 = ["Monday", "Thursday", "Tuesday", "Sunday"]
print (list1, len(list1), type(list1), id(list1))
list1.append("Saturday")
print (list1, len(list1), type(list1), id(list1))
list1.append("Friday")
```

```
print (list1, len(list1), type(list1), id(list1))
list1.append("Wednesday")
print (list1, len(list1), type(list1), id(list1))
```

```
['Monday', 'Thursday', 'Tuesday', 'Sunday'] 4 <class 'list'> 2716496553984
['Monday', 'Thursday', 'Tuesday', 'Sunday', 'Saturday'] 5 <class 'list'> 2716496553984
['Monday', 'Thursday', 'Tuesday', 'Sunday', 'Saturday', 'Friday'] 6 <class 'list'> 2716496553984
['Monday', 'Thursday', 'Tuesday', 'Sunday', 'Saturday', 'Friday', 'Wednesday'] 7 <class 'list'> 2716496553984
```

In [151…

```
list1 = ["Monday", "Thursday", "Tuesday", "Sunday"]
print (list1, len(list1), type(list1), id(list1))
list1.insert(2, "Saturday")
print (list1, len(list1), type(list1), id(list1))
list1.insert(4, "Friday")
print (list1, len(list1), type(list1), id(list1))
list1.insert(1, "Wednesday")
print (list1, len(list1), type(list1), id(list1))
list1.insert(100, "Weekday")
print (list1, len(list1), type(list1), id(list1))
```

```
['Monday', 'Thursday', 'Tuesday', 'Sunday'] 4 <class 'list'> 2716493376768
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Sunday'] 5 <class 'list'> 2716493376768
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716493376768
['Monday', 'Wednesday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 7 <class 'list'> 2716493376768
['Monday', 'Wednesday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday', 'Weekday'] 8 <class 'list'> 271649337
6768
```

In [153…

```
# delete opeation on list
list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday']
print (list1, len(list1), type(list1), id(list1))
list1.clear()
print (list1, len(list1), type(list1), id(list1))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716503464896
[] 0 <class 'list'> 2716503464896
```

In [154…

```
list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday']
print (list1, len(list1), type(list1), id(list1))
del list1
print (list1, len(list1), type(list1), id(list1))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716496488640

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-154-7f9073c1b939> in <module>
      2 print (list1, len(list1), type(list1), id(list1))
      3 del list1
----> 4 print (list1, len(list1), type(list1), id(list1))

NameError: name 'list1' is not defined
```

In [160…

```
list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday']
print (list1, len(list1), type(list1), id(list1))
list1.remove('Saturday')
print (list1, len(list1), type(list1), id(list1))
list1.remove('Sunday')
print (list1, len(list1), type(list1), id(list1))
list1.remove('Saturday')
print (list1, len(list1), type(list1), id(list1))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716502828544
['Monday', 'Thursday', 'Tuesday', 'Friday', 'Sunday'] 5 <class 'list'> 2716502828544
['Monday', 'Thursday', 'Tuesday', 'Friday'] 4 <class 'list'> 2716502828544

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-160-48a263281054> in <module>
      5 list1.remove('Sunday')
      6 print (list1, len(list1), type(list1), id(list1))
----> 7 list1.remove('Saturday')
      8 print (list1, len(list1), type(list1), id(list1))

ValueError: list.remove(x): x not in list
```

In [158…

```
# update opeation on list
list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday']
print (list1, len(list1), type(list1), id(list1))
list1[3] = "Wednesday"
print (list1, len(list1), type(list1), id(list1))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716498046784
['Monday', 'Thursday', 'Saturday', 'Wednesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716498046784
```

In [162…

```
list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday']
print (list1, len(list1), type(list1), id(list1))
list2 = ['Friday', 'Sunday']
print (list2, len(list2), type(list2), id(list2))
result = list1 + list2   # list concatenation
print (result, len(result), type(result), id(result))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday'] 4 <class 'list'> 2716502490880
['Friday', 'Sunday'] 2 <class 'list'> 2716495608576
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716503374528
```

In [164…

```
list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday']
print (list1, len(list1), type(list1), id(list1))
list2 = ['Friday', 'Sunday']
print (list2, len(list2), type(list2), id(list2))
list1.extend(list2)   # extending the list
print (list1, len(list1), type(list1), id(list1))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday'] 4 <class 'list'> 2716502478784
['Friday', 'Sunday'] 2 <class 'list'> 2716500713408
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716502478784
```

In [165…

```
list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday']
print (list1, len(list1), type(list1), id(list1))
list2 = ['Friday', 'Sunday']
print (list2, len(list2), type(list2), id(list2))
result = [list1, list2]   # list of lists
print (result, len(result), type(result), id(result))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday'] 4 <class 'list'> 2716498787968
['Friday', 'Sunday'] 2 <class 'list'> 2716498787200
[['Monday', 'Thursday', 'Saturday', 'Tuesday'], ['Friday', 'Sunday']] 2 <class 'list'> 2716502478784
```

In [171…

```
list1 = [['Monday', 'Thursday', 'Saturday', 'Tuesday'], ['Friday', 'Sunday']]
print (list1, len(list1), type(list1), id(list1))
print (list1[0][1], list1[-2][-3], list1[1][1], list1[-1][-1])
print (list1[0][2][2:5], list1[-2][-2][-6:-3])
```

```
[['Monday', 'Thursday', 'Saturday', 'Tuesday'], ['Friday', 'Sunday']] 2 <class 'list'> 2716500944704
Thursday Thursday Sunday Sunday
tur tur
```

In [179…

```
list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday']
print (list1, len(list1), type(list1), id(list1))
list1.sort()
print (list1, len(list1), type(list1), id(list1))
list1.sort(reverse = True)
print (list1, len(list1), type(list1), id(list1))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716492762048
['Friday', 'Monday', 'Saturday', 'Sunday', 'Thursday', 'Tuesday'] 6 <class 'list'> 2716492762048
['Tuesday', 'Thursday', 'Sunday', 'Saturday', 'Monday', 'Friday'] 6 <class 'list'> 2716492762048
```

In [181…

```
list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday']
print (list1, len(list1), type(list1), id(list1))
result = sorted(list1)
print (list1, len(list1), type(list1), id(list1))
print (result, len(result), type(result), id(result))
result = sorted(list1, reverse = True)
print (result, len(result), type(result), id(result))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716503085184
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716503085184
['Friday', 'Monday', 'Saturday', 'Sunday', 'Thursday', 'Tuesday'] 6 <class 'list'> 2716503756544
['Tuesday', 'Thursday', 'Sunday', 'Saturday', 'Monday', 'Friday'] 6 <class 'list'> 2716503776512
```

In [183…

```
list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Saturday', 'Friday', 'Sunday', 'Saturday']
print (list1, len(list1), type(list1), id(list1))
print (list1.count('Saturday'), list1.count('Tuesday'))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Saturday', 'Friday', 'Sunday', 'Saturday'] 8 <class 'list'> 271649491
0592
3 1
```

In [188…

```
list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Saturday', 'Friday', 'Sunday', 'Saturday']
print (list1, len(list1), type(list1), id(list1))
print (list1.index('Saturday'))
print (list1.index('Saturday', 0))
print (list1.index('Saturday', 3))
print (list1.index('Saturday', 5))
print (list1.index('Saturday', 8))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Saturday', 'Friday', 'Sunday', 'Saturday'] 8 <class 'list'> 271649468
2496
2
2
4
7
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-188-103aa24113cb> in <module>
      5 print (list1.index('Saturday', 3))
      6 print (list1.index('Saturday', 5))
----> 7 print (list1.index('Saturday', 8))

ValueError: 'Saturday' is not in list
```

In [189…

```
list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday']
print (list1, len(list1), type(list1), id(list1))
print (list1.pop())
print (list1, len(list1), type(list1), id(list1))
print (list1.pop())
print (list1, len(list1), type(list1), id(list1))
print (list1.pop())
print (list1, len(list1), type(list1), id(list1))
print (list1.pop())
print (list1, len(list1), type(list1), id(list1))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716496148736
Sunday
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday'] 5 <class 'list'> 2716496148736
Friday
['Monday', 'Thursday', 'Saturday', 'Tuesday'] 4 <class 'list'> 2716496148736
Tuesday
['Monday', 'Thursday', 'Saturday'] 3 <class 'list'> 2716496148736
Saturday
['Monday', 'Thursday'] 2 <class 'list'> 2716496148736
```

In [216…

```
list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday']
print (list1, len(list1), type(list1), id(list1))
print (list1.pop(1))
print (list1, len(list1), type(list1), id(list1))
print (list1.pop(3))
print (list1, len(list1), type(list1), id(list1))
print (list1.pop(2))
print (list1, len(list1), type(list1), id(list1))
print (list1.pop(0))
print (list1, len(list1), type(list1), id(list1))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716497419776
Thursday
['Monday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 5 <class 'list'> 2716497419776
Friday
['Monday', 'Saturday', 'Tuesday', 'Sunday'] 4 <class 'list'> 2716497419776
Tuesday
['Monday', 'Saturday', 'Sunday'] 3 <class 'list'> 2716497419776
Monday
['Saturday', 'Sunday'] 2 <class 'list'> 2716497419776
```

In [208…

```
list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday']
print (list1, len(list1), type(list1), id(list1))
list2 = list1
print (list2, len(list2), type(list2), id(list2))
list1[0] = 'Wednesday'
print (list1, len(list1), type(list1), id(list1))
print (list2, len(list2), type(list2), id(list2))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716474996800
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716474996800
['Wednesday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716474996800
['Wednesday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716474996800
```

In [210…

```
list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday']
print (list1, len(list1), type(list1), id(list1))
list2[:] = list1
print (list2, len(list2), type(list2), id(list2))
list1[0] = 'Wednesday'
print (list1, len(list1), type(list1), id(list1))
print (list2, len(list2), type(list2), id(list2))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716497494784
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716474996800
['Wednesday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716497494784
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716474996800
```

In [211…

```
list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday']
```

```
print (list1, len(list1), type(list1), id(list1))
list2 = list1.copy()
print (list2, len(list2), type(list2), id(list2))
list1[0] = 'Wednesday'
print (list1, len(list1), type(list1), id(list1))
print (list2, len(list2), type(list2), id(list2))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716495690816
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716494007232
['Wednesday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716495690816
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716494007232
```

In [218…

```
list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday']
print (list1, len(list1), type(list1), id(list1))
print (all(list1))
list1 = ['Monday', 'Thursday', False, 'Saturday', 'Tuesday', 'Friday', 'Sunday']
print (list1, len(list1), type(list1), id(list1))
print (all(list1))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716493337984
True
['Monday', 'Thursday', False, 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 7 <class 'list'> 2716498711744
False
```

In [221…

```
list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday']
print (list1, len(list1), type(list1), id(list1))
print (any(list1))
list1 = ['Monday', 'Thursday', False, 'Saturday', 'Tuesday', 'Friday', 'Sunday']
print (list1, len(list1), type(list1), id(list1))
print (any(list1))
list1 = [False, False, False]
print (list1, len(list1), type(list1), id(list1))
print (any(list1))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716495679616
True
['Monday', 'Thursday', False, 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 7 <class 'list'> 2716499643136
True
[False, False, False] 3 <class 'list'> 2716497418432
False
```

In [212…

```
def myfunct(mylist):
    mylist[3] = 'Wednesday'

list1 = ['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday']
print (list1, len(list1), type(list1), id(list1))
myfunct(list1)
print (list1, len(list1), type(list1), id(list1))
```

```
['Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716498342784
['Monday', 'Thursday', 'Saturday', 'Wednesday', 'Friday', 'Sunday'] 6 <class 'list'> 2716498342784
```

Dealing with the Tuple Processing

> Tuple is a collection of data items of same or different datatypes enclosed with in first brackets. Tuples items are immutable as INSERT, DELETE and UPDATE operations can not be carried out on them.

In [193…

```
tuple1 = (100, 500, 200, 400, 300)
print (tuple1, len(tuple1), type(tuple1), id(tuple1))
print (max(tuple1), min(tuple1))
print (sum(tuple1), sum(tuple1) / len(tuple1))
```

```
(100, 500, 200, 400, 300) 5 <class 'tuple'> 2716495852064
500 100
1500 300.0
```

In [194…

```
tuple1 = ("Monday", "Friday", "Thursday", "Tuesday", "Sunday")
print (tuple1, len(tuple1), type(tuple1), id(tuple1))
print (max(tuple1), min(tuple1))
print (sum(tuple1), sum(tuple1) / len(tuple1))
```

```
('Monday', 'Friday', 'Thursday', 'Tuesday', 'Sunday') 5 <class 'tuple'> 2716492705200
Tuesday Friday
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-194-ec338ebb34a5> in <module>
      2 print (tuple1, len(tuple1), type(tuple1), id(tuple1))
      3 print (max(tuple1), min(tuple1))
----> 4 print (sum(tuple1), sum(tuple1) / len(tuple1))

TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

In [195…

```
tuple1 = ("Monday", 900, "Friday", "Thursday", 700, "Tuesday", "Sunday")
print (tuple1, len(tuple1), type(tuple1), id(tuple1))
print (max(tuple1), min(tuple1))
print (sum(tuple1), sum(tuple1) / len(tuple1))
```

```
('Monday', 900, 'Friday', 'Thursday', 700, 'Tuesday', 'Sunday') 7 <class 'tuple'> 2716495264352
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-195-65ea8b8c3fc0> in <module>
      1 tuple1 = ("Monday", 900, "Friday", "Thursday", 700, "Tuesday", "Sunday")
      2 print (tuple1, len(tuple1), type(tuple1), id(tuple1))
----> 3 print (max(tuple1), min(tuple1))
      4 print (sum(tuple1), sum(tuple1) / len(tuple1))

TypeError: '>' not supported between instances of 'int' and 'str'
```

In [196…

```
tuple1 = (100, 500.45, 200, False, 400.5, True, 300)
print (tuple1, len(tuple1), type(tuple1), id(tuple1))
print (max(tuple1), min(tuple1))
print (sum(tuple1), sum(tuple1) / len(tuple1))
```

```
(100, 500.45, 200, False, 400.5, True, 300) 7 <class 'tuple'> 2716498639552
500.45 False
1501.95 214.56428571428572
```

In [ ]:

```
# tuple indexing and slicing
index from left to right ->   0        1          2         3           4
                  tuple1 -> ("Sunday" "Tuesday", "Friday", "Saturday", "Thursday")
index from right to left ->  -5       -4         -3        -2          -1
```

In [197…

```
tuple1 = ("Sunday", "Tuesday", "Friday", "Saturday", "Thursday")
print (tuple1[2], tuple1[-3], tuple1[4], tuple1[-1])
print (tuple1[0::2], tuple1[1::2], tuple1[::-1])
print (tuple1[3][2:5], tuple1[-2][-6:-3])
```

```
Friday Friday Thursday Thursday
('Sunday', 'Friday', 'Thursday') ('Tuesday', 'Saturday') ('Thursday', 'Saturday', 'Friday', 'Tuesday', 'Sunday')
tur tur
```

In [199…

```
tuple1 = ('Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday')
print (tuple1, len(tuple1), type(tuple1), id(tuple1))
del tuple1
print (tuple1, len(tuple1), type(tuple1), id(tuple1))
```

```
('Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday') 6 <class 'tuple'> 2716492692160
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-199-09f2e9bf08f2> in <module>
      2 print (tuple1, len(tuple1), type(tuple1), id(tuple1))
      3 del tuple1
----> 4 print (tuple1, len(tuple1), type(tuple1), id(tuple1))

NameError: name 'tuple1' is not defined
```

In [200…

```
tuple1 = ('Monday', 'Thursday', 'Saturday', 'Tuesday')
print (tuple1, len(tuple1), type(tuple1), id(tuple1))
tuple2 = ('Friday', 'Sunday')
print (tuple2, len(tuple2), type(tuple2), id(tuple2))
result = tuple1 + tuple2   # tuple concatenation
print (result, len(result), type(result), id(result))
```

```
('Monday', 'Thursday', 'Saturday', 'Tuesday') 4 <class 'tuple'> 2716504136336
('Friday', 'Sunday') 2 <class 'tuple'> 2716499545472
('Monday', 'Thursday', 'Saturday', 'Tuesday', 'Friday', 'Sunday') 6 <class 'tuple'> 2716495262816
```

In [201…

```
tuple1 = ('Monday', 'Thursday', 'Saturday', 'Tuesday')
print (tuple1, len(tuple1), type(tuple1), id(tuple1))
tuple2 = ('Friday', 'Sunday')
print (tuple2, len(tuple2), type(tuple2), id(tuple2))
result = (tuple1, tuple2)   # tuple of tuples
print (result, len(result), type(result), id(result))
```

```
('Monday', 'Thursday', 'Saturday', 'Tuesday') 4 <class 'tuple'> 2716502450640
('Friday', 'Sunday') 2 <class 'tuple'> 2716496762496
(('Monday', 'Thursday', 'Saturday', 'Tuesday'), ('Friday', 'Sunday')) 2 <class 'tuple'> 2716499147456
```

In [202…

```
tuple1 = (('Monday', 'Thursday', 'Saturday', 'Tuesday'), ('Friday', 'Sunday'))
print (tuple1, len(tuple1), type(tuple1), id(tuple1))
print (tuple1[0][1], tuple1[-2][-3], tuple1[1][1], tuple1[-1][-1])
print (tuple1[0][2][2:5], tuple1[-2][-2][-6:-3])
```

```
(('Monday', 'Thursday', 'Saturday', 'Tuesday'), ('Friday', 'Sunday')) 2 <class 'tuple'> 2716497258368
Thursday Thursday Sunday Sunday
tur tur
```

In [203…

```
tuple1 = ('Monday', 'Thursday', 'Saturday', 'Tuesday', 'Saturday', 'Friday', 'Sunday', 'Saturday')
print (tuple1, len(tuple1), type(tuple1), id(tuple1))
print (tuple1.count('Saturday'), tuple1.count('Tuesday'))
```

```
('Monday', 'Thursday', 'Saturday', 'Tuesday', 'Saturday', 'Friday', 'Sunday', 'Saturday') 8 <class 'tuple'> 27164949
27456
3 1
```

In [204…

```
tuple1 = ('Monday', 'Thursday', 'Saturday', 'Tuesday', 'Saturday', 'Friday', 'Sunday', 'Saturday')
print (tuple1, len(tuple1), type(tuple1), id(tuple1))
print (tuple1.index('Saturday'))
print (tuple1.index('Saturday', 0))
print (tuple1.index('Saturday', 3))
print (tuple1.index('Saturday', 5))
print (tuple1.index('Saturday', 8))
```

```
('Monday', 'Thursday', 'Saturday', 'Tuesday', 'Saturday', 'Friday', 'Sunday', 'Saturday') 8 <class 'tuple'> 27164921
33984
2
2
4
7
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-204-bcfb046ab5fa> in <module>
      5 print (tuple1.index('Saturday', 3))
      6 print (tuple1.index('Saturday', 5))
----> 7 print (tuple1.index('Saturday', 8))

ValueError: tuple.index(x): x not in tuple
```

In [206…

```
tuple1 = ('Monday', 'Thursday', 'Saturday', 'Tuesday')
print (tuple1, len(tuple1), type(tuple1), id(tuple1))
# tuple1[2] = 'Sunday'   # ERROR !!!
list1 = list(tuple1)
print (list1, len(list1), type(list1), id(list1))
list1[2] = 'Sunday'
print (list1, len(list1), type(list1), id(list1))
tuple1 = tuple(list1)
print (tuple1, len(tuple1), type(tuple1), id(tuple1))
```

```
('Monday', 'Thursday', 'Saturday', 'Tuesday') 4 <class 'tuple'> 2716502947456
['Monday', 'Thursday', 'Saturday', 'Tuesday'] 4 <class 'list'> 2716493107392
['Monday', 'Thursday', 'Sunday', 'Tuesday'] 4 <class 'list'> 2716493107392
('Monday', 'Thursday', 'Sunday', 'Tuesday') 4 <class 'tuple'> 2716495855184
```

Dealing with the Dictionary Processing

> Dictionary is a collection of key-value pairs, where key should be always immutable type and value may be immutable or mutable types. Dictionary items are enclosed with in second brackets (curly braces). Dictionary items are mutable as INSERT, DELETE and UPDATE operations can be carried out on them.

In [215…

```
dict1 = {"mango":"green", "orange":"orange", "guava":"green"}
print (dict1, len(dict1), type(dict1), id(dict1))
print (dict1.items())   # outcome is in the list of tuples
print (dict1.keys())   # outcome is in the list
print (dict1.values())   # outcome is in the list
```

```
{'mango': 'green', 'orange': 'orange', 'guava': 'green'} 3 <class 'dict'> 2716499214080
dict_items([('mango', 'green'), ('orange', 'orange'), ('guava', 'green')])
dict_keys(['mango', 'orange', 'guava'])
dict_values(['green', 'orange', 'green'])
```

In [223…

```
# accessing value against the key
dict1 = {"mango":"green", "orange":"orange", "guava":"green"}
print (dict1, len(dict1), type(dict1), id(dict1))
print (dict1.get("orange"))
print (dict1.get("guava"))
print (dict1.get("guava", "not found..."))
print (dict1.get("apple"))
print (dict1.get("apple", "not found..."))
```

```
{'mango': 'green', 'orange': 'orange', 'guava': 'green'} 3 <class 'dict'> 2716492594304
orange
green
green
```

```
None
not found...
```

In [225…

```
dict1 = {"mango":"green", "orange":"orange", "guava":"green"}
print (dict1, len(dict1), type(dict1), id(dict1))
print (dict1["orange"])
print (dict1["guava"])
print (dict1["apple"])
```

```
{'mango': 'green', 'orange': 'orange', 'guava': 'green'} 3 <class 'dict'> 2716495018304
orange
green
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-225-769dffa4acac> in <module>
      3 print (dict1["orange"])
      4 print (dict1["guava"])
----> 5 print (dict1["apple"])

KeyError: 'apple'
```

In [228…

```
# insert and update operations
dict1 = {"mango":"green", "orange":"orange", "guava":"green"}
print (dict1, len(dict1), type(dict1), id(dict1))
dict1["pineapple"] = "yellow"   # insert
print (dict1, len(dict1), type(dict1), id(dict1))
dict1["apple"] = "red"   # insert
print (dict1, len(dict1), type(dict1), id(dict1))
dict1["mango"] = "red"   # update
print (dict1, len(dict1), type(dict1), id(dict1))
```

```
{'mango': 'green', 'orange': 'orange', 'guava': 'green'} 3 <class 'dict'> 2716499183936
{'mango': 'green', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow'} 4 <class 'dict'> 2716499183936
{'mango': 'green', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow', 'apple': 'red'} 5 <class 'dict'> 271
6499183936
{'mango': 'red', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow', 'apple': 'red'} 5 <class 'dict'> 27164
99183936
```

In [230…

```
# delete operation
dict1 = {'mango': 'red', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow', 'apple': 'red'}
print (dict1, len(dict1), type(dict1), id(dict1))
dict1.clear()
print (dict1, len(dict1), type(dict1), id(dict1))
```

```
{'mango': 'red', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow', 'apple': 'red'} 5 <class 'dict'> 27164
95983232
{} 0 <class 'dict'> 2716495983232
```

In [231…

```
dict1 = {'mango': 'red', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow', 'apple': 'red'}
print (dict1, len(dict1), type(dict1), id(dict1))
del dict1
print (dict1, len(dict1), type(dict1), id(dict1))
```

```
{'mango': 'red', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow', 'apple': 'red'} 5 <class 'dict'> 27164
96082304
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-231-66f69df5a7c6> in <module>
      2 print (dict1, len(dict1), type(dict1), id(dict1))
      3 del dict1
----> 4 print (dict1, len(dict1), type(dict1), id(dict1))

NameError: name 'dict1' is not defined
```

In [233…

```
dict1 = {'mango': 'red', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow', 'apple': 'red'}
print (dict1, len(dict1), type(dict1), id(dict1))
print (dict1.pop("orange"))
print (dict1, len(dict1), type(dict1), id(dict1))
print (dict1.pop("apple"))
print (dict1, len(dict1), type(dict1), id(dict1))
print (dict1.pop("pineapple"))
print (dict1, len(dict1), type(dict1), id(dict1))
print (dict1.pop("coconut"))
print (dict1, len(dict1), type(dict1), id(dict1))
```

```
{'mango': 'red', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow', 'apple': 'red'} 5 <class 'dict'> 27164
95622592
orange
{'mango': 'red', 'guava': 'green', 'pineapple': 'yellow', 'apple': 'red'} 4 <class 'dict'> 2716495622592
red
{'mango': 'red', 'guava': 'green', 'pineapple': 'yellow'} 3 <class 'dict'> 2716495622592
yellow
{'mango': 'red', 'guava': 'green'} 2 <class 'dict'> 2716495622592
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-233-13c3aae8ef31> in <module>
      7 print (dict1.pop("pineapple"))
      8 print (dict1, len(dict1), type(dict1), id(dict1))
----> 9 print (dict1.pop("coconut"))
     10 print (dict1, len(dict1), type(dict1), id(dict1))

KeyError: 'coconut'
```

In [236…

```
dict1 = {'mango': 'red', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow', 'apple': 'red'}
print (dict1, len(dict1), type(dict1), id(dict1))
print (dict1.popitem())
print (dict1, len(dict1), type(dict1), id(dict1))
print (dict1.popitem())
print (dict1, len(dict1), type(dict1), id(dict1))
print (dict1.popitem())
print (dict1, len(dict1), type(dict1), id(dict1))
print (dict1.popitem())
print (dict1, len(dict1), type(dict1), id(dict1))
```

```
{'mango': 'red', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow', 'apple': 'red'} 5 <class 'dict'> 27164
95726528
('apple', 'red')
{'mango': 'red', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow'} 4 <class 'dict'> 2716495726528
('pineapple', 'yellow')
{'mango': 'red', 'orange': 'orange', 'guava': 'green'} 3 <class 'dict'> 2716495726528
('guava', 'green')
{'mango': 'red', 'orange': 'orange'} 2 <class 'dict'> 2716495726528
('orange', 'orange')
{'mango': 'red'} 1 <class 'dict'> 2716495726528
```

In [240…

```
dict1 = {'mango': 'red', 'orange': 'orange', 'guava': 'green'}
print (dict1, len(dict1), type(dict1), id(dict1))
dict2 = {'pineapple': 'yellow', 'apple': 'red', 'mango': 'green'}
print (dict2, len(dict2), type(dict2), id(dict2))
dict1.update(dict2)   # dictionary concatenation
print (dict1, len(dict1), type(dict1), id(dict1))
print (dict2, len(dict2), type(dict2), id(dict2))
```

```
{'mango': 'red', 'orange': 'orange', 'guava': 'green'} 3 <class 'dict'> 2716498711360
{'pineapple': 'yellow', 'apple': 'red', 'mango': 'green'} 3 <class 'dict'> 2716498713408
{'mango': 'green', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow', 'apple': 'red'} 5 <class 'dict'> 271
6498711360
{'pineapple': 'yellow', 'apple': 'red', 'mango': 'green'} 3 <class 'dict'> 2716498713408
```

In [248…

```
fruits = ['mango', 'orange', 'guava', 'pineapple', 'apple', "banana"]
colors = ['green', 'orange', 'green', 'yellow', 'red']
print (fruits, len(fruits))
print (colors, len(colors))
result = zip(fruits, colors)
print (result, type(result))
print ()
result = list(zip(fruits, colors))
print (result, type(result))
result = dict(list(zip(fruits, colors)))
print (result, type(result))
print ()
result = tuple(zip(fruits, colors))
print (result, type(result))
result = dict(list(zip(fruits, colors)))
print (result, type(result))
```

```
['mango', 'orange', 'guava', 'pineapple', 'apple', 'banana'] 6
['green', 'orange', 'green', 'yellow', 'red'] 5
<zip object at 0x000002787C390E40> <class 'zip'>

[('mango', 'green'), ('orange', 'orange'), ('guava', 'green'), ('pineapple', 'yellow'), ('apple', 'red')] <class 'li
st'>
{'mango': 'green', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow', 'apple': 'red'} <class 'dict'>

(('mango', 'green'), ('orange', 'orange'), ('guava', 'green'), ('pineapple', 'yellow'), ('apple', 'red')) <class 'tu
ple'>
{'mango': 'green', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow', 'apple': 'red'} <class 'dict'>
```

In [250…

```
mykeys = ['key-1', 'key-2', 'key-3', 'key-4']
myvalue = 100
dict1 = dict.fromkeys(mykeys, myvalue)
print (dict1, len(dict1))
```

```
{'key-1': 100, 'key-2': 100, 'key-3': 100, 'key-4': 100} 4
```

In [254…

```
dict1 = {'mango': 'red', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow'}
print (dict1, len(dict1), type(dict1), id(dict1))
```

```
dict1.setdefault('mango', 'green')
print (dict1, len(dict1), type(dict1), id(dict1))
dict1.setdefault('pineapple', 'light yellow')
print (dict1, len(dict1), type(dict1), id(dict1))
dict1.setdefault('apple', 'red')
print (dict1, len(dict1), type(dict1), id(dict1))
```

```
{'mango': 'red', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow'} 4 <class 'dict'> 2716495245568
{'mango': 'red', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow'} 4 <class 'dict'> 2716495245568
{'mango': 'red', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow'} 4 <class 'dict'> 2716495245568
{'mango': 'red', 'orange': 'orange', 'guava': 'green', 'pineapple': 'yellow', 'apple': 'red'} 5 <class 'dict'> 27164
95245568
```

CLASS ASSIGNMENT-3 / Day-2

> Take a sentence with upper and lower case letters and other characters from the user and find frequency of each alphabet in the sentence. Consider upper and lower letters are the same letters.

In [255…

```
sentence = input("Please enter a sentence: ").lower()
charfreq = {}
for ch in sentence:
    if (ch.isalpha()):
        if (ch in charfreq):
            charfreq[ch] += 1
        else:
            charfreq[ch] = 1
print ("So required frequency of characters is:", charfreq)
print ("End of the program...")
```

```
So required frequency of characters is: {'r': 1, 'a': 2, 'm': 1, 'i': 1, 's': 1, 'g': 1, 'o': 3, 'd': 1, 'b': 1,
'y': 1}
End of the program...
```

Dealing with the Set Processing

> Set is a collection of unique unordered values of same or different datatypes enclosed within second brackets (curly braces). Set items are mutable as INSERT, DELETE and UPDATE operations can be carried out on them.

In [260…

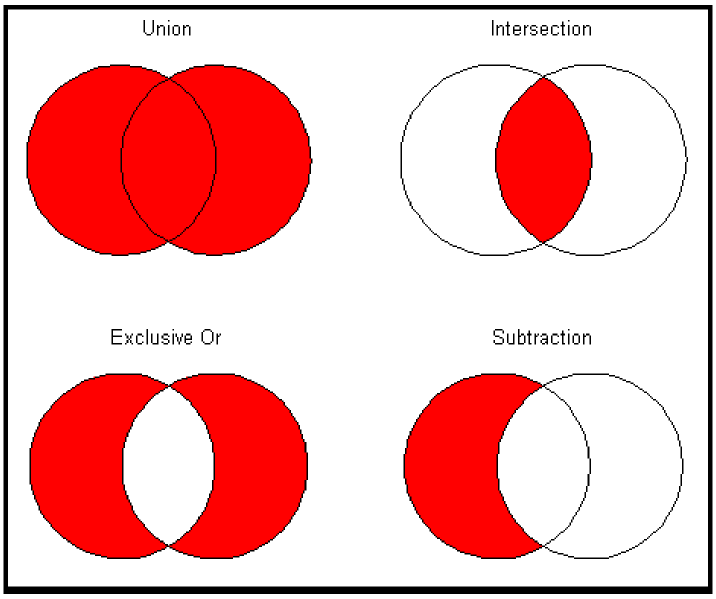
```
# forming a set from a list
languages = set(['c++', 'kotlin', 'java', 'perl'])
print (languages, len(languages), type(languages), id(languages))
print()
# forming a set from a tuple
languages = set(('c++', 'java', 'perl', 'kotlin'))
print (languages, len(languages), type(languages), id(languages))
print()
# forming a set directly
languages = {'c++', 'java', 'perl', 'kotlin'}
print (languages, len(languages), type(languages), id(languages))
```

```
{'c++', 'kotlin', 'java', 'perl'} 4 <class 'set'> 2716498246560

{'c++', 'java', 'kotlin', 'perl'} 4 <class 'set'> 2716506114304

{'c++', 'java', 'kotlin', 'perl'} 4 <class 'set'> 2716498246112
```

Set Operations: Union, Intersection, Exclusive Or (Symmetric Difference), Subtraction (Set Difference)



In [266…

```
# set union operation
languages = {'c++', 'python', 'java'}
print (languages, len(languages), type(languages), id(languages))
snakes = {'cobra', 'viper', 'python'}
print (snakes, len(snakes), type(snakes), id(snakes))
result = languages.union(snakes)
print (result, len(result), type(result), id(result))
result = languages | snakes
print (result, len(result), type(result), id(result))
```

```
{'c++', 'java', 'python'} 3 <class 'set'> 2716505059392
{'viper', 'cobra', 'python'} 3 <class 'set'> 2716499078272
{'viper', 'c++', 'java', 'cobra', 'python'} 5 <class 'set'> 2716505062080
{'viper', 'c++', 'java', 'cobra', 'python'} 5 <class 'set'> 2716505063200
```

In [271…

```
# set intersection operation
languages = {'c++', 'python', 'java'}
print (languages, len(languages), type(languages), id(languages))
snakes = {'cobra', 'viper', 'python'}
print (snakes, len(snakes), type(snakes), id(snakes))
result = languages.intersection(snakes)
print (result, len(result), type(result), id(result))
result = languages & snakes
print (result, len(result), type(result), id(result))
languages.intersection_update(snakes)
print (languages, len(languages), type(languages), id(languages))
print (snakes, len(snakes), type(snakes), id(snakes))
```

```
{'c++', 'java', 'python'} 3 <class 'set'> 2716501142432
{'viper', 'cobra', 'python'} 3 <class 'set'> 2716505019552
{'python'} 1 <class 'set'> 2716505018880
{'python'} 1 <class 'set'> 2716505021568
{'python'} 1 <class 'set'> 2716501142432
{'viper', 'cobra', 'python'} 3 <class 'set'> 2716505019552
```

In [274…

```
# set difference operation
languages = {'c++', 'python', 'java'}
print (languages, len(languages), type(languages), id(languages))
snakes = {'cobra', 'viper', 'python'}
print (snakes, len(snakes), type(snakes), id(snakes))
result = languages.difference(snakes)
print (result, len(result), type(result), id(result))
result = languages - snakes
print (result, len(result), type(result), id(result))
languages.difference_update(snakes)
print (languages, len(languages), type(languages), id(languages))
print (snakes, len(snakes), type(snakes), id(snakes))
```

```
{'c++', 'java', 'python'} 3 <class 'set'> 2716499430528
{'viper', 'cobra', 'python'} 3 <class 'set'> 2716499428064
{'c++', 'java'} 2 <class 'set'> 2716499427616
{'c++', 'java'} 2 <class 'set'> 2716499429632
{'c++', 'java'} 2 <class 'set'> 2716499430528
{'viper', 'cobra', 'python'} 3 <class 'set'> 2716499428064
```

In [276…

```
# set symmetric difference operation
languages = {'c++', 'python', 'java'}
print (languages, len(languages), type(languages), id(languages))
```

```
snakes = {'cobra', 'viper', 'python'}
print (snakes, len(snakes), type(snakes), id(snakes))
result = languages.symmetric_difference(snakes)
print (result, len(result), type(result), id(result))
result = languages ^ snakes
print (result, len(result), type(result), id(result))
languages.symmetric_difference_update(snakes)
print (languages, len(languages), type(languages), id(languages))
print (snakes, len(snakes), type(snakes), id(snakes))
```

```
{'c++', 'java', 'python'} 3 <class 'set'> 2716498507584
{'viper', 'cobra', 'python'} 3 <class 'set'> 2716498509376
{'c++', 'viper', 'java', 'cobra'} 4 <class 'set'> 2716498506688
{'c++', 'viper', 'java', 'cobra'} 4 <class 'set'> 2716498508928
{'c++', 'viper', 'java', 'cobra'} 4 <class 'set'> 2716498507584
{'viper', 'cobra', 'python'} 3 <class 'set'> 2716498509376
```

In [281…

```
set1 = {100, 200, 300, 400, 500}
set2 = {100, 300, 400}
set3 = {600, 700, 800, 900}
print (set1.issubset(set2), set2.issubset(set1), set1.issuperset(set2), set2.issuperset(set1))
print (set1.isdisjoint(set2), set1.isdisjoint(set3))
```

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

In [283…

```
languages = {'c++', 'python', 'java'}
print (languages, len(languages), type(languages), id(languages))
languages.add('kotlin')
print (languages, len(languages), type(languages), id(languages))
languages.add('perl')
print (languages, len(languages), type(languages), id(languages))
languages.add('python')
print (languages, len(languages), type(languages), id(languages))
languages.add('c')
print (languages, len(languages), type(languages), id(languages))
languages.add('kotlin')
print (languages, len(languages), type(languages), id(languages))
```

```
{'c++', 'java', 'python'} 3 <class 'set'> 2716502586528
{'c++', 'java', 'kotlin', 'python'} 4 <class 'set'> 2716502586528
{'kotlin', 'c++', 'python', 'java', 'perl'} 5 <class 'set'> 2716502586528
{'kotlin', 'c++', 'python', 'java', 'perl'} 5 <class 'set'> 2716502586528
{'kotlin', 'c++', 'python', 'c', 'java', 'perl'} 6 <class 'set'> 2716502586528
{'kotlin', 'c++', 'python', 'c', 'java', 'perl'} 6 <class 'set'> 2716502586528
```

In [287…

```
languages1 = {'c++', 'python', 'java'}
print (languages1, len(languages1), type(languages1), id(languages1))
languages2 = {'c', 'perl', 'golang', 'julia'}
print (languages2, len(languages2), type(languages2), id(languages2))
languages1.update(languages2)   # set concatenation
print (languages1, len(languages1), type(languages1), id(languages1))
print (languages2, len(languages2), type(languages2), id(languages2))
```

```
{'c++', 'java', 'python'} 3 <class 'set'> 2716504016480
{'golang', 'c', 'julia', 'perl'} 4 <class 'set'> 2716504016928
{'julia', 'c', 'golang', 'c++', 'java', 'python', 'perl'} 7 <class 'set'> 2716504016480
{'golang', 'c', 'julia', 'perl'} 4 <class 'set'> 2716504016928
```

In [289…

```
languages = {'julia', 'c', 'golang', 'c++', 'java', 'python', 'perl'}
print (languages, len(languages), type(languages), id(languages))
print (languages.pop())
print (languages, len(languages), type(languages), id(languages))
print (languages.pop())
print (languages, len(languages), type(languages), id(languages))
print (languages.pop())
print (languages, len(languages), type(languages), id(languages))
print (languages.pop())
print (languages, len(languages), type(languages), id(languages))
```

```
{'golang', 'c++', 'python', 'julia', 'c', 'java', 'perl'} 7 <class 'set'> 2716501162240
golang
{'c++', 'python', 'julia', 'c', 'java', 'perl'} 6 <class 'set'> 2716501162240
c++
{'python', 'julia', 'c', 'java', 'perl'} 5 <class 'set'> 2716501162240
python
{'julia', 'c', 'java', 'perl'} 4 <class 'set'> 2716501162240
julia
{'c', 'java', 'perl'} 3 <class 'set'> 2716501162240
```

In [290…

```
languages = {'julia', 'c', 'golang', 'c++', 'java', 'python', 'perl'}
print (languages, len(languages), type(languages), id(languages))
languages.remove('c')
```

```
print (languages, len(languages), type(languages), id(languages))
languages.remove('java')
print (languages, len(languages), type(languages), id(languages))
languages.remove('perl')
print (languages, len(languages), type(languages), id(languages))
languages.remove('perl')
print (languages, len(languages), type(languages), id(languages))
```

```
{'golang', 'c++', 'python', 'julia', 'c', 'java', 'perl'} 7 <class 'set'> 2716499172704
{'golang', 'c++', 'python', 'julia', 'java', 'perl'} 6 <class 'set'> 2716499172704
{'golang', 'c++', 'python', 'julia', 'perl'} 5 <class 'set'> 2716499172704
{'golang', 'c++', 'python', 'julia'} 4 <class 'set'> 2716499172704
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-290-0a26b60538c2> in <module>
      7 languages.remove('perl')
      8 print (languages, len(languages), type(languages), id(languages))
----> 9 languages.remove('perl')
     10 print (languages, len(languages), type(languages), id(languages))

KeyError: 'perl'
```

In [294…

```
languages = {'julia', 'c', 'golang', 'c++', 'java', 'python', 'perl'}
print (languages, len(languages), type(languages), id(languages))
languages.discard('c')
print (languages, len(languages), type(languages), id(languages))
languages.discard('java')
print (languages, len(languages), type(languages), id(languages))
languages.discard('perl')
print (languages, len(languages), type(languages), id(languages))
languages.discard('perl')
print (languages, len(languages), type(languages), id(languages))
```

```
{'golang', 'c++', 'python', 'julia', 'c', 'java', 'perl'} 7 <class 'set'> 2716495737120
{'golang', 'c++', 'python', 'julia', 'java', 'perl'} 6 <class 'set'> 2716495737120
{'golang', 'c++', 'python', 'julia', 'perl'} 5 <class 'set'> 2716495737120
{'golang', 'c++', 'python', 'julia'} 4 <class 'set'> 2716495737120
{'golang', 'c++', 'python', 'julia'} 4 <class 'set'> 2716495737120
```

In [291…

```
languages = {'julia', 'c', 'golang', 'c++', 'java', 'python', 'perl'}
print (languages, len(languages), type(languages), id(languages))
languages.clear()
print (languages, len(languages), type(languages), id(languages))
```

```
{'golang', 'c++', 'python', 'julia', 'c', 'java', 'perl'} 7 <class 'set'> 2716498533280
set() 0 <class 'set'> 2716498533280
```

In [292…

```
languages = {'julia', 'c', 'golang', 'c++', 'java', 'python', 'perl'}
print (languages, len(languages), type(languages), id(languages))
del languages
print (languages, len(languages), type(languages), id(languages))
```

```
{'golang', 'c++', 'python', 'julia', 'c', 'java', 'perl'} 7 <class 'set'> 2716498534176
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-292-5cdbbbcf16dc> in <module>
      2 print (languages, len(languages), type(languages), id(languages))
      3 del languages
----> 4 print (languages, len(languages), type(languages), id(languages))

NameError: name 'languages' is not defined
```

In [293…

```
languages = {'julia', 'c', 'golang', 'c++', 'java', 'python', 'perl'}
print (languages, len(languages), type(languages), id(languages))
languages.remove("python")
languages.add("python 3.10")
print (languages, len(languages), type(languages), id(languages))
```

```
{'golang', 'c++', 'python', 'julia', 'c', 'java', 'perl'} 7 <class 'set'> 2716505022240
{'golang', 'c++', 'julia', 'c', 'java', 'python 3.10', 'perl'} 7 <class 'set'> 2716505022240
```

Dealing with the Frozen-Set Processing

> Frozen-Set is a collection of unique unordered values of same or different datatypes enclosed within second brackets (curly braces). Frozen-Set items are immutable as INSERT, DELETE and UPDATE operations can not be carried out on them.

In [261…

```
# forming a frozen-set from a list
languages = frozenset(['c++', 'kotlin', 'java', 'perl'])
print (languages, len(languages), type(languages), id(languages))
print()
```

```
# forming a set from a tuple
languages = frozenset(('c++', 'java', 'perl', 'kotlin'))
print (languages, len(languages), type(languages), id(languages))
```

```
frozenset({'c++', 'kotlin', 'java', 'perl'}) 4 <class 'frozenset'> 2716506114304

frozenset({'c++', 'java', 'kotlin', 'perl'}) 4 <class 'frozenset'> 2716498246560
```

In [267…

```
# frozen-set union operation
languages = frozenset(['c++', 'python', 'java'])
print (languages, len(languages), type(languages), id(languages))
snakes = frozenset(['cobra', 'viper', 'python'])
print (snakes, len(snakes), type(snakes), id(snakes))
result = languages.union(snakes)
print (result, len(result), type(result), id(result))
result = languages | snakes
print (result, len(result), type(result), id(result))
```

```
frozenset({'c++', 'java', 'python'}) 3 <class 'frozenset'> 2716501062592
frozenset({'viper', 'cobra', 'python'}) 3 <class 'frozenset'> 2716501063488
frozenset({'viper', 'c++', 'java', 'cobra', 'python'}) 5 <class 'frozenset'> 2716501062144
frozenset({'viper', 'c++', 'java', 'cobra', 'python'}) 5 <class 'frozenset'> 2716501063040
```

In [273…

```
# frozen-set intersection operation
languages = frozenset(['c++', 'python', 'java'])
print (languages, len(languages), type(languages), id(languages))
snakes = frozenset(['cobra', 'viper', 'python'])
print (snakes, len(snakes), type(snakes), id(snakes))
result = languages.intersection(snakes)
print (result, len(result), type(result), id(result))
result = languages & snakes
print (result, len(result), type(result), id(result))
```

```
frozenset({'c++', 'java', 'python'}) 3 <class 'frozenset'> 2716492792992
frozenset({'viper', 'cobra', 'python'}) 3 <class 'frozenset'> 2716496150592
frozenset({'python'}) 1 <class 'frozenset'> 2716496188128
frozenset({'python'}) 1 <class 'frozenset'> 2716496191040
```

In [275…

```
# frozen-set difference operation
languages = frozenset(['c++', 'python', 'java'])
print (languages, len(languages), type(languages), id(languages))
snakes = frozenset(['cobra', 'viper', 'python'])
print (snakes, len(snakes), type(snakes), id(snakes))
result = languages.difference(snakes)
print (result, len(result), type(result), id(result))
result = languages - snakes
print (result, len(result), type(result), id(result))
```

```
frozenset({'c++', 'java', 'python'}) 3 <class 'frozenset'> 2716495825376
frozenset({'viper', 'cobra', 'python'}) 3 <class 'frozenset'> 2716495826720
frozenset({'c++', 'java'}) 2 <class 'frozenset'> 2716495826272
frozenset({'c++', 'java'}) 2 <class 'frozenset'> 2716495737568
```

In [277…

```
# frozen-set symmetric difference operation
languages = frozenset(['c++', 'python', 'java'])
print (languages, len(languages), type(languages), id(languages))
snakes = frozenset(['cobra', 'viper', 'python'])
print (snakes, len(snakes), type(snakes), id(snakes))
result = languages.symmetric_difference(snakes)
print (result, len(result), type(result), id(result))
result = languages ^ snakes
print (result, len(result), type(result), id(result))
```

```
frozenset({'c++', 'java', 'python'}) 3 <class 'frozenset'> 2716495826272
frozenset({'viper', 'cobra', 'python'}) 3 <class 'frozenset'> 2716506180512
frozenset({'c++', 'viper', 'java', 'cobra'}) 4 <class 'frozenset'> 2716503445792
frozenset({'c++', 'viper', 'java', 'cobra'}) 4 <class 'frozenset'> 2716503445568
```

In [282…

```
frozenset1 = frozenset((100, 200, 300, 400, 500))
frozenset2 = frozenset([100, 300, 400])
frozenset3 = frozenset([600, 700, 800, 900])
print (frozenset1.issubset(frozenset2), frozenset2.issubset(frozenset1), frozenset1.issuperset(frozenset2), frozense
print (frozenset1.isdisjoint(frozenset2), frozenset1.isdisjoint(frozenset3))
```

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

Python Data File Handling

In [295…

```
# importing required modules
import csv
```

In [325…

```
# with open('emp_data.csv') as data_file:
# with open('C://Users//Arnab//USA Batch//Vodafone95//emp_data.csv') as data_file:
# with open('C:/Users/Arnab/USA Batch/Vodafone95/emp_data.csv') as data_file:
with open('C:\\Users\\Arnab\\USA Batch\Vodafone95\emp_data.csv') as data_file:
    csv_reader = csv.reader(data_file)
    print (csv_reader)
    print (list(csv_reader))
    print (len(list(csv_reader)))
```

```
<_csv.reader object at 0x000002787BACDFA0>
[['1001', 'Dhiman', 'Kolkata', '39000'], ['1002', 'Anupam', 'Kolkata', '25000'], ['1003', 'Subham', 'Mumbai', '3600
0'], ['1004', 'Dinesh', 'Chennai', '28000'], ['1005', 'Kakali', 'Mumbai', '25000'], ['1006', 'Bimal', 'Hyderabad',
'30000'], ['1007', 'Tarun', 'Chennai', '17000'], ['1008', 'Rittik', 'Durgapur', '45000'], ['1009', 'Barun', 'Hyderab
ad', '39000'], ['1010', 'Utpal', 'Lucknow', '20000']]
0
```

In [326…

```
with open('emp_data.csv') as data_file:
    csv_reader = csv.reader(data_file)
    for row in csv_reader:
        print (f"Emp-ID: {row[0]}, Emp-Name: {row[1]}, Emp-Loc: {row[2]}, Emp-Salary: {row[3]}...")
```

```
Emp-ID: 1001, Emp-Name: Dhiman, Emp-Loc: Kolkata, Emp-Salary: 39000...
Emp-ID: 1002, Emp-Name: Anupam, Emp-Loc: Kolkata, Emp-Salary: 25000...
Emp-ID: 1003, Emp-Name: Subham, Emp-Loc: Mumbai, Emp-Salary: 36000...
Emp-ID: 1004, Emp-Name: Dinesh, Emp-Loc: Chennai, Emp-Salary: 28000...
Emp-ID: 1005, Emp-Name: Kakali, Emp-Loc: Mumbai, Emp-Salary: 25000...
Emp-ID: 1006, Emp-Name: Bimal, Emp-Loc: Hyderabad, Emp-Salary: 30000...
Emp-ID: 1007, Emp-Name: Tarun, Emp-Loc: Chennai, Emp-Salary: 17000...
Emp-ID: 1008, Emp-Name: Rittik, Emp-Loc: Durgapur, Emp-Salary: 45000...
Emp-ID: 1009, Emp-Name: Barun, Emp-Loc: Hyderabad, Emp-Salary: 39000...
Emp-ID: 1010, Emp-Name: Utpal, Emp-Loc: Lucknow, Emp-Salary: 20000...
```

In [ ]: