# Tuple

# Tuple Creation

In [5]:
```
tuple1 = () # empty
```

In [8]:
```
tuple2 = (10,20,30)  # tuple of integers number
```

In [9]:
```
tuple3 = (10.77,30.66,60.89) # tuple of float numbers
```

In [10]:
```
tuple4 = ('one','two','three') # tuple of string
```

In [11]:
```
tuple5 = ('python ai',25,(50,100),(150,90)) # Nested Tuple
```

In [12]:
```
tuple6 = (100,'python ML',17.765) # Tuple of Mixed Data Type
```

In [14]:
```
tuple7 = ('python ds ',25,[50,100],[150,90],{'Ajay', 'Adam'},(99,22,33))
```

In [16]:
```
len(tuple7) # lenth of list
```

Out[16]: 6

# Tuple Indexing

In [18]:
```
tuple1
```

Out[18]: ()

In [20]:
```
tuple2[0] # retreive frist element of tuple
```

Out[20]: 10

In [22]:
```
tuple4[0] # Retreive first element of he tuple
```

Out[22]: 'one'

In [23]:
```
tuple4[0][0] # Nested indexing - Access the first chsrscter of the firest tuple elm
```

Out[23]: 'o'

In [24]:
```
tuple4[-1] # Last item of the tuple
```

Out[24]: 'three'

In [25]:
```
tuple5[-1] # Last Iteam of the Tuple
```

Out[25]:
```
(150, 90)
```

# Tuple Slicing

In [26]:
```
mytuple = ('one','two','three','four','five','six','seven','eight')
```

In [27]:
```
mytuple[0:3 # Return all iteams from 0th to 3rd index location excluding the items
```

Out[27]:
```
('one', 'two', 'three')
```

In [29]:
```
mytuple[2:5] # List all items from 2nd to 5th index locationexcluding the items of
```

Out[29]:
```
('three', 'four', 'five')
```

In [31]:
```
mytuple[:3] # Return first three items
```

Out[31]:
```
('one', 'two', 'three')
```

In [33]:
```
mytuple[:2] # Returns first two items
```

Out[33]:
```
('one', 'two')
```

In [35]:
```
mytuple[-3:] # Return last three items
```

Out[35]:
```
('six', 'seven', 'eight')
```

In [36]:
```
mytuple[-2:] # returns last two items
```

Out[36]:
```
('seven', 'eight')
```

In [38]:
```
mytuple[-1:] # Returs last one item of the tuple
```

Out[38]:
```
('eight',)
```

In [39]:
```
mytuple[:] # Return whole tuple
```

Out[39]:
```
('one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight')
```

# Tuple Remove & Change Items

In [40]:
```
mytuple
```

Out[40]:
```
('one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight')
```

In [41]:
```
del mytuple[0] # # Tuple are immutable which means we can not DELET tuple items
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[41], line 1
----> 1 del mytuple[0]

TypeError: 'tuple' object doesn't support item deletion
```

In [42]:
```
mytuple[0] = 1 # Tuple are immutable which means we can not CHANGE items
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[42], line 1
----> 1 mytuple[0] = 1

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

In [43]:
```
del mytuple # Deleting entire object is possible
```

In [44]:
```
mytuple
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[44], line 1
----> 1 mytuple

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

In [45]:
```
mytuple = ('one','two','three','four','five','six','seven','eight')
```

In [46]:
```
mytuple
```

Out[46]:
```
('one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight')
```

# Loop Through a Tuple

In [47]:
```
for i in mytuple:
    print(i)
```

```
one
two
three
four
five
six
seven
eight
```

In [48]:
```
for i in enumerate(mytuple):
    print(i)
```

```
(0, 'one')
(1, 'two')
(2, 'three')
(3, 'four')
(4, 'five')
(5, 'six')
(6, 'seven')
(7, 'eight')
```

# Tuple Membership

In [49]:
```
mytuple
```

Out[49]: ('one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight')

In [50]:
```
'one'in mytuple # check if 'one' exist in the list
```

Out[50]: True

In [51]:
```
'seven' in mytuple
```

Out[51]: True

In [52]:
```
'ten' in mytuple # Check if 'ten' exist in the list
```

Out[52]: False

In [54]:
```
if 'three' in mytuple:  # Check if 'three' in the list
    print('Three is present in the tuple')
else:
    prin('Three is not present in the tuple')
```

```
Three is present in the tuple
```

In [55]:
```
if 'eleven' in mytuple: # Check if 'eleven' in the list
    print('Eleven is present in the tuple')
else:
    print('Eleven is not present in the tuple')
```

```
Eleven is not present in the tuple
```

# Index Position

In [56]:
```
mytuple
```

Out[56]: ('one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight')

In [57]:
```
mytuple.index('one') # Index of first element equal to 'one'
```

Out[57]: 0

In [59]:
```
mytuple.index('five') # Index of first element equal to 'five'
```

Out[59]: 4

In [61]:
```
mytuple1 = ('one', 'two', 'three', 'four', 'one', 'six', 'one', 'eight')
```

In [62]:
```
mytuple1
```

Out[62]: ('one', 'two', 'three', 'four', 'one', 'six', 'one', 'eight')

In [63]:
```
mytuple1.index('one') # Index of first element equal to 'one'
```

Out[63]: 0

# Sorting

In [64]:
```
mytuple2 = (43,67,99,12,6,90,97)
```

In [65]:
```
mytuple2
```

Out[65]: (43, 67, 99, 12, 6, 90, 97)

In [68]:
```
sorted(mytuple2) # RETRUNS A NEW SORTED AND DOES't Change original tuple
```

Out[68]: [6, 12, 43, 67, 90, 97, 99]

In [67]:
```
sorted(mytuple2, reverse=True) # sort in descending oder
```

Out[67]: [99, 97, 90, 67, 43, 12, 6]

# Set

In [70]:
```
s = {}
s
```

Out[70]: {}

In [71]:
```
type(s)
```

Out[71]: dict

In [73]:
```
s1 = set()
s1
```

Out[73]: set()

In [74]:
```
s2 = {90,10,50,40,25,10,50} # duplicate values not allowed
s2
```

Out[74]: {10, 25, 40, 50, 90}

In [75]:
```
type(s2)
```

Out[75]: set

In [76]:
```
s2
```

Out[76]: {10, 25, 40, 50, 90}

In [77]:
```
s3 = s2.copy() # unable to copy elements from set to se
s3
```

Out[77]: {10, 25, 40, 50, 90}

In [78]:
```
s3
```

Out[78]: {10, 25, 40, 50, 90}

In [79]:
```
s3.add(3.4) # element is addin set
```

In [80]:
```
s3
```

Out[80]: {3.4, 10, 25, 40, 50, 90}

In [81]:
```
s3.add('Python') # In set add string
```

In [82]:
```
s3
```

Out[82]: {10, 25, 3.4, 40, 50, 90, 'Python'}

In [83]:
```
s3.add(1+2j) # added difrrent values in set
s3.add(True)
```

In [84]:
```
s3
```

Out[84]: {(1+2j), 10, 25, 3.4, 40, 50, 90, 'Python', True}

In [85]:
```
print(s)
print(s1)
print(s2)
print(s3)
```

```
{}
set()
{50, 90, 40, 25, 10}
{'Python', True, 3.4, (1+2j), 10, 25, 90, 40, 50}
```

In [86]:
```
s
```

Out[86]: {}

In [87]:
```
type(s)
```

Out[87]:
```
dict
```

In [88]:
```
s2
```

Out[88]:
```
{10, 25, 40, 50, 90}
```

In [89]:
```
s3
```

Out[89]:
```
{(1+2j), 10, 25, 3.4, 40, 50, 90, 'Python', True}
```

In [90]:
```
s3.remove(2000)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[90], line 1
----> 1 s3.remove(2000)

KeyError: 2000
```

In [91]:
```
s3
```

Out[91]:
```
{(1+2j), 10, 25, 3.4, 40, 50, 90, 'Python', True}
```

In [92]:
```
s3.remove(1+2j)
```

In [93]:
```
s3
```

Out[93]:
```
{10, 25, 3.4, 40, 50, 90, 'Python', True}
```

In [94]:
```
s3.discard(10)
```

In [95]:
```
s3
```

Out[95]:
```
{25, 3.4, 40, 50, 90, 'Python', True}
```

In [96]:
```
s3.discard(2000)
```

In [97]:
```
s3
```

Out[97]:
```
{25, 3.4, 40, 50, 90, 'Python', True}
```

In [98]:
```
s3.pop()
```

Out[98]:
```
'Python'
```

In [99]:
```
s3.pop()
```

Out[99]:
```
True
```

In [100…
```
s3.pop()
```

Out[100… 3.4

In [101…
```
s3
```

Out[101… {25, 40, 50, 90}

In [103…
```
s3.pop(0)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[103], line 1
----> 1 s3.pop(0)

TypeError: set.pop() takes no arguments (1 given)
```

In [104…
```
s3
```

Out[104… {25, 40, 50, 90}

In [105…
```
s3[:]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[105], line 1
----> 1 s3[:]

TypeError: 'set' object is not subscriptable
```

In [106…
```
s3
```

Out[106… {25, 40, 50, 90}

In [107…
```
s3[1:]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[107], line 1
----> 1 s3[1:]

TypeError: 'set' object is not subscriptable
```

In [108…
```
s3
```

Out[108… {25, 40, 50, 90}

In [109…
```
s3[2]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[109], line 1
----> 1 s3[2]

TypeError: 'set' object is not subscriptable
```

In [110…
```
s3
```

Out[110…
```
{25, 40, 50, 90}
```

In [111…
```
s3.pop(0)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[111], line 1
----> 1 s3.pop(0)

TypeError: set.pop() takes no arguments (1 given)
```

In [112…
```
s3.pop()
```

Out[112…
```
25
```

In [113…
```
s3
```

Out[113…
```
{40, 50, 90}
```

In [114…
```
40 in s3
```

Out[114…
```
True
```

# Set Operation

In [115…
```
a = {1,2,3,4,5,}
b = {4,5,6,7,8}
c = {8,9,10}
```

In [116…
```
type(c)
```

Out[116…
```
set
```

In [117…
```
a.union(b)
```

Out[117…
```
{1, 2, 3, 4, 5, 6, 7, 8}
```

In [118…
```
a.union(b,c)
```

Out[118…
```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
```

In [119…
```
print(a)
print(b)
print(c)
```

```
{1, 2, 3, 4, 5}
{4, 5, 6, 7, 8}
{8, 9, 10}
```

In [120…
```
a | b
```
Out[120… {1, 2, 3, 4, 5, 6, 7, 8}

In [121…
```
b | c
```
Out[121… {4, 5, 6, 7, 8, 9, 10}

In [122…
```
a | b | c
```
Out[122… {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

In [123…
```
a|c
```
Out[123… {1, 2, 3, 4, 5, 8, 9, 10}

In [124…
```
a|c|b
```
Out[124… {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

# Intersection

In [125…
```
a = {1,2,3,4,5,}
b = {4,5,6,7,8}
c = {8,9,10}
```

In [126…
```
a.intersection(b)
```
Out[126… {4, 5}

In [127…
```
b.intersection(c)
```
Out[127… {8}

In [128…
```
a & b
```
Out[128… {4, 5}

In [129…
```
b & c
```
Out[129… {8}

In [130…
```
a & b
```
Out[130… {4, 5}

In [131…
```
b & c
```
Out[131… {8}

# Differncece

In [132…
```
a = {1,2,3,4,5,}
b = {4,5,6,7,8}
c = {8,9,10}
```

In [133…
```
a.difference(b)
```

Out[133… {1, 2, 3}

In [134…
```
b.difference(a)
```

Out[134… {6, 7, 8}

In [135…
```
a = {1,2,3,4,5,}
b = {4,5,6,7,8}
c = {8,9,10}
```

In [136…
```
b -c
```

Out[136… {4, 5, 6, 7}

In [137…
```
c - b
```

Out[137… {9, 10}

In [138…
```
a -b - c
```

Out[138… {1, 2, 3}

In [ ]: