Tuple Creation

```
In [4]:  tup1 = ()    #Empty Tuple
```

```
In [6]:  tup2 = (10,30,60)  #Tuple of Integer numbers
```

```
In [8]:  tup3 = (10.77, 30.66, 60.89)    #Tuple of Float numbers
```

```
In [10]: tup4 = ('one', 'two', "three")     #Tuple of Strings
```

```
In [12]: tup5 = ('Asif', 25, (50, 100), (150, 90))  #Nested Tuples
```

```
In [14]: tup6 = (100, 'Asif', 17.765)  #Tuple of Mixed Data Types
```

```
In [16]: tup7 = ('Asif', 25, [50, 100], [150, 90], {'John', 'David'}, (99, 22, 33))
```

```
In [18]: len(tup7)  #Length of list

Out[18]: 6
```

Tuple Indexing

```
In [21]: tup2[0]  #Retrieves first element of tuple

Out[21]: 10
```

```
In [23]: tup4[0]  #Retrieves first element of tuple

Out[23]: 'one'
```

```
In [27]: tup4[0][0]  #Nested Indexing - Access the first character of the first Tuple ele

Out[27]: 'o'
```

```
In [29]: tup4[-1]  #Last item of the tuple

Out[29]: 'three'
```

```
In [31]: tup5[-1]  #Last item of the tuple

Out[31]: (150, 90)
```

Tuple Slicing

```
In [34]: mytuple = ('one', 'two', 'three', 'four', 'five', 'six', 'seven', 'Eight')
```

```
In [38]: mytuple[0][2] #Nested Indexing - Access the n-1th character of the first index e
```

```
Out[38]: 'e'
```

```
In [42]: mytuple[0:3] #Returns all items from 0th to 3rd index, excluding the item(as rig
```

```
Out[42]: ('one', 'two', 'three')
```

```
In [44]: mytuple[2:5]
```

```
Out[44]: ('three', 'four', 'five')
```

```
In [46]: mytuple[:3]  #Return first 3 items
```

```
Out[46]: ('one', 'two', 'three')
```

```
In [48]: mytuple[:2]  #Return first 2 items
```

```
Out[48]: ('one', 'two')
```

```
In [52]: mytuple[:-3]  #Return all element before last 3 items
```

```
Out[52]: ('one', 'two', 'three', 'four', 'five')
```

```
In [54]: mytuple[-3]  #Return -3th index element
```

```
Out[54]: 'six'
```

```
In [56]: mytuple[-3:]  #Return last 3 items
```

```
Out[56]: ('six', 'seven', 'Eight')
```

```
In [58]: mytuple[-2:]  #Return last 2 items
```

```
Out[58]: ('seven', 'Eight')
```

```
In [64]: mytuple[-1:]  #Return last item
```

```
Out[64]: ('Eight',)
```

```
In [66]: mytuple[-1]  #Return last item
```

```
Out[66]: 'Eight'
```

```
In [68]: mytuple[:]  #Return whole tuple
```

```
Out[68]: ('one', 'two', 'three', 'four', 'five', 'six', 'seven', 'Eight')
```

Remove and Change Items

```
In [71]: mytuple
```

```
Out[71]: ('one', 'two', 'three', 'four', 'five', 'six', 'seven', 'Eight')
```

```
In [75]: del mytuple[0]  #Tuples are Immutable i.e We can't DELETE Tuple items
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[75], line 1
----> 1 del mytuple[0]

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

In [77]:
```
mytuple[0] = 1  #Tuples are Immutable i.e We can't CHANGE Tuple items
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[77], line 1
----> 1 mytuple[0] = 1

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

In [79]:
```
del mytuple  #Deleting entire Tuple elements is Possible
```

In [81]:
```
mytuple  # Gives error as tuple is deleted
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[81], line 1
----> 1 mytuple

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

Loop Through a Tuple

In [5]:
```
mytuple = ('one', 'two', 'three', 'four', 'five', 'six', 'seven', 'Eight')
```

In [88]:
```
mytuple
```

Out[88]:
```
('one', 'two', 'three', 'four', 'five', 'six', 'seven', 'Eight')
```

In [90]:
```
for i in mytuple:
    print (i)
```

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

In [92]:
```
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 [7]:
```
mytuple
```

Out[7]:
```
('one', 'two', 'three', 'four', 'five', 'six', 'seven', 'Eight')
```

In [97]:
```
'one' in mytuple  #Check if 'one' exists in the list
```

Out[97]:
```
True
```

In [99]:
```
'ten' in mytuple  #Check if 'ten' exists in the list
```

Out[99]:
```
False
```

In [9]:
```
if 'three' in mytuple:   #Check if 'three' exists in the list
    print('three is present in the tuple')
else:
    print('three is not present in the tuple')
```

```
three is present in the tuple
```

In [11]:
```
if 'eleven' in mytuple:   #Check if 'three' exists 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 [14]:
```
mytuple
```

Out[14]:
```
('one', 'two', 'three', 'four', 'five', 'six', 'seven', 'Eight')
```

In [22]:
```
mytuple.index('one') #Prints the index of 'one' in the tuple
```

Out[22]:
```
0
```

In [26]:
```
mytuple.index('five') #Prints the index of 'five' in the tuple
```

Out[26]:
```
4
```

In [32]:
```
mytuple1 = ('one', 'two', 'three', 'four', 'one', 'one', 'two', 'three')
mytuple1
```

```
Out[32]: ('one', 'two', 'three', 'four', 'one', 'one', 'two', 'three')
```

```
In [36]: mytuple1.index('one')  #Prints the index of 'one' in the tuple
```

```
Out[36]: 0
```

Sorting

```
In [39]: mytuple2 = (43, 67, 99, 12, 6, 90, 67)
```

```
In [41]: sorted(mytuple2)  #Returns a new sorted list and doesn't change the original tup
```

```
Out[41]: [6, 12, 43, 67, 67, 90, 99]
```

```
In [45]: sorted(mytuple2, reverse=True)  #Sort in descending order
```

```
Out[45]: [99, 90, 67, 67, 43, 12, 6]
```

```
In [ ]:
```