Pre-defined functions in Tuple-(Immutable):

--> We know that object of tuple we can perfor both indexing and slicing operations and along with these operations: 1).index() 2).count()

The functions are not present in tuple

1).append() 2).insert() 3).remove() 4).clear() 5).pop() 6).pop(index) 7).reverse() 8).sort() 9).copy() 10).extend()

In [5]:
```
tpl=(10,"Khan",56.40,True,2+3j,20)
print(tpl,type(tpl))
```
```
(10, 'Khan', 56.4, True, (2+3j), 20) <class 'tuple'>
```

In [13]:
```
tpl.count("Khan")
```
Out[13]:
```
1
```

In [15]:
```
tpl.index("Khan")
```
Out[15]:
```
1
```

In [34]:
```
tpl=(10,"Khan",56.40,True,2+3j,20)
lst=list(tpl)
print(lst,type(lst))
```
```
[10, 'Khan', 56.4, True, (2+3j), 20] <class 'list'>
```

In [36]:
```
tpl=(10,20,30,-10,-20,-30)
lst=list(tpl)
print(lst,type(lst))
```
```
[10, 20, 30, -10, -20, -30] <class 'list'>
```

In [38]:
```
lst.sort()
print(lst)
```
```
[-30, -20, -10, 10, 20, 30]
```

In [42]:
```
tpl=tuple(lst)
print(tpl)
```
```
(-30, -20, -10, 10, 20, 30)
```

In [46]:
```
s="MISSISSIPPI"
print(s)
```
```
MISSISSIPPI
```

In [48]:
```
s.sort # not possible
print(s)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[48], line 1
----> 1 s.sort # not possible
      2 print(s)

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

In [50]:
```
x=sorted(s)
print(x)
```

```
['I', 'I', 'I', 'I', 'M', 'P', 'P', 'S', 'S', 'S', 'S']
```

In [52]:
```
"".join(x)
```

Out[52]:
```
'IIIIMPPSSSS'
```

In [54]:
```
help(tuple) #It displays a detataild description of tuples including their methods
```

```
Help on class tuple in module builtins:

class tuple(object)
 |  tuple(iterable=(), /)
 |
 |  Built-in immutable sequence.
 |
 |  If no argument is given, the constructor returns an empty tuple.
 |  If iterable is specified the tuple is initialized from iterable's items.
 |
 |  If the argument is a tuple, the return value is the same object.
 |
 |  Built-in subclasses:
 |      asyncgen_hooks
 |      MonthDayNano
 |      UnraisableHookArgs
 |
 |  Methods defined here:
 |
 |  __add__(self, value, /)
 |      Return self+value.
 |
 |  __contains__(self, key, /)
 |      Return bool(key in self).
 |
 |  __eq__(self, value, /)
 |      Return self==value.
 |
 |  __ge__(self, value, /)
 |      Return self>=value.
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __getitem__(self, key, /)
 |      Return self[key].
 |
 |  __getnewargs__(self, /)
 |
 |  __gt__(self, value, /)
 |      Return self>value.
 |
 |  __hash__(self, /)
 |      Return hash(self).
 |
 |  __iter__(self, /)
 |      Implement iter(self).
 |
 |  __le__(self, value, /)
 |      Return self<=value.
 |
 |  __len__(self, /)
 |      Return len(self).
 |
 |  __lt__(self, value, /)
 |      Return self<value.
 |
 |  __mul__(self, value, /)
 |      Return self*value.
 |
 |  __ne__(self, value, /)
 |      Return self!=value.
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __rmul__(self, value, /)
 |      Return value*self.
 |
 |  count(self, value, /)
 |      Return number of occurrences of value.
 |
 |  index(self, value, start=0, stop=9223372036854775807, /)
 |      Return first index of value.
 |
 |      Raises ValueError if the value is not present.
 |
 |  ----------------------------------------------------------------------
 |  Class methods defined here:
 |
 |  __class_getitem__(...)
 |      See PEP 585
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs)
 |      Create and return a new object.  See help(type) for accurate signature.
```

In [60]:
```
# case 1: tuple in tupe is possible
tpl=(10,20,30,-10,-20,-30)
print(tpl,type(tpl))
```

```
(10, 20, 30, -10, -20, -30) <class 'tuple'>
```

In [64]:
```
for val in tpl:
    print(val,"-->",type(val),"-->",type(tpl))
```

```
10 --> <class 'int'> --> <class 'tuple'>
20 --> <class 'int'> --> <class 'tuple'>
30 --> <class 'int'> --> <class 'tuple'>
-10 --> <class 'int'> --> <class 'tuple'>
-20 --> <class 'int'> --> <class 'tuple'>
-30 --> <class 'int'> --> <class 'tuple'>
```

In [66]:
```
# case 2: tuple in list is possible
lst=[10,"Mahaboob",(20,30,40),(50,60,70),"MRIIRS"]
print(lst,type(lst))
```

```
[10, 'Mahaboob', (20, 30, 40), (50, 60, 70), 'MRIIRS'] <class 'list'>
```

```
In [68]: for val in lst:
             print(val,"-->",type(val),"-->",type(tpl))
```

```
10 --> <class 'int'> --> <class 'tuple'>
Mahaboob --> <class 'str'> --> <class 'tuple'>
(20, 30, 40) --> <class 'tuple'> --> <class 'tuple'>
(50, 60, 70) --> <class 'tuple'> --> <class 'tuple'>
MRIIRS --> <class 'str'> --> <class 'tuple'>
```

```
In [72]: # case 3: list in tuple is possible
         tpl=(10,20,30,-10,-20,-30)
         print(tpl,type(tpl))
```

```
(10, 20, 30, -10, -20, -30) <class 'tuple'>
```

```
In [74]: for val in tpl:
             print(val,"-->",type(val),"-->",type(tpl))
```

```
10 --> <class 'int'> --> <class 'tuple'>
20 --> <class 'int'> --> <class 'tuple'>
30 --> <class 'int'> --> <class 'tuple'>
-10 --> <class 'int'> --> <class 'tuple'>
-20 --> <class 'int'> --> <class 'tuple'>
-30 --> <class 'int'> --> <class 'tuple'>
```

Set Category Data Types

```
In [ ]: # The purpose of set catogery data type is that to store multiple values or either
        # In the Python programming we have two types of set category data types
        # 1).set() --> (mutable and immutable)
        # 2).frozenset() --> (immutable)
```

```
In [83]: s={10,20,30,10,20,40,50,60,70}
         print(s,type(s))
```

```
{50, 20, 70, 40, 10, 60, 30} <class 'set'>
```

```
In [92]: # set never mainted of insertion order because PVM displys any one of the possiblit
         # we can't perform index and slicing operations because set object does not maintai
         #
```

```
In [96]: s={10,20,30,10,20,40,50,60,70}
         print(s,type(s))
```

```
{50, 20, 70, 40, 10, 60, 30} <class 'set'>
```

```
In [98]: s[0]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[98], line 1
----> 1 s[0]

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

In [100…
```
s[0:3]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[100], line 1
----> 1 s[0:3]

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

In [104…
```
s1=[10,"Mahaboob",30,2+3j,True,20.34,"MRIIRS"]
print(s1,type(s1))
```

```
[10, 'Mahaboob', 30, (2+3j), True, 20.34, 'MRIIRS'] <class 'list'>
```

In [123…
```
s="Mahaboob Khan"
print(s,type(s))
```

```
Mahaboob Khan <class 'str'>
```

In [125…
```
s1=set(s)
print(s1,type(s1))
```

```
{'h', 'M', 'n', ' ', 'K', 'b', 'o', 'a'} <class 'set'>
```

In [133…
```
lst=[10,20,30,10,20,30,10,20,30] # set is removing duplicates
s1=set(lst)
print(s1,type(s1))
```

```
{10, 20, 30} <class 'set'>
```

In [ ]:
```
                                                # END #
```