Tuple

1. Tuple is similar to List except that the objects in tuple are immutable which means we cannot change the elements of a tuple once assigned. 2. When we do not want to change the data over time, tuple is a preferred data type. 3. Iterating over the elements of a tuple is faster compared to iterating over a list.

In [3]:
```
t1=()
type(t1)
```

Out[3]:
```
tuple
```

In [4]:
```
t1
```

Out[4]:
```
()
```

In [5]:
```
t2=(10,30,60) #tuple of integers
t2
```

Out[5]:
```
(10, 30, 60)
```

In [6]:
```
t3=(10.77,30.66,60.89) #tuple of float
t3
```

Out[6]:
```
(10.77, 30.66, 60.89)
```

t4=('one','two','three') #tuple of string t4

In [9]:
```
t5=('amit',25,(50,100),(150,90)) #Nested tuple
t5
```

Out[9]:
```
('amit', 25, (50, 100), (150, 90))
```

In [10]:
```
 t6=('amit',20.41,78,7+5j,True,85) #tuple of mixed data type
t6
```

Out[10]:
```
('amit', 20.41, 78, (7+5j), True, 85)
```

In [11]:
```
len(t6)
```

Out[11]: 6

tuple indexing

In [13]:
```
t6[0]
```

Out[13]: 'amit'

In [14]:
```
t6[-1]
```

Out[14]: 85

In [15]:
```
t6[:]
```

Out[15]: ('amit', 20.41, 78, (7+5j), True, 85)

In [16]:
```
t6[0][1]
```

Out[16]: 'm'

tuple slicing

In [17]:
```
t7 = ('sixty one' , 'sixty two' , 'sixty three' , 'sixty four' , 'sixty five' , 'sixty six' , 'sixty seven' , 'sixty
t7
```

Out[17]:
```
('sixty one',
 'sixty two',
 'sixty three',
 'sixty four',
 'sixty five',
 'sixty six',
 'sixty seven',
 'sixty eight')
```

In [18]:
```
t7[::-1]
```

Out[18]:
```
('sixty eight',
 'sixty seven',
 'sixty six',
 'sixty five',
 'sixty four',
 'sixty three',
 'sixty two',
 'sixty one')
```

In [19]:
```
t7[-1]
```

Out[19]:
```
'sixty eight'
```

In [20]:
```
t7[0:6]
```

Out[20]:
```
('sixty one',
 'sixty two',
 'sixty three',
 'sixty four',
 'sixty five',
 'sixty six')
```

In [21]:
```
t7[:3]
```

Out[21]:
```
('sixty one', 'sixty two', 'sixty three')
```

In [22]:
```
t7[3:]
```

Out[22]:
```
('sixty four', 'sixty five', 'sixty six', 'sixty seven', 'sixty eight')
```

In [23]:
```
t7[-3:]
```

Out[23]:
```
('sixty six', 'sixty seven', 'sixty eight')
```

In [24]:
```
t7[0:9:2]
```

Out[24]:
```
('sixty one', 'sixty three', 'sixty five', 'sixty seven')
```

In [27]:
```
t8=('amit',25,(50,100),(150,90)) #Delete entire tuple object is possible
t8
```

Out[27]: ('amit', 25, (50, 100), (150, 90))

In [28]:
```
del t8
```

In [29]:
```
t8
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[29], line 1
----> 1 t8

NameError: name 't8' is not defined
```

loop through a tuple

In [30]:
```
t7
```

Out[30]:
```
('sixty one',
 'sixty two',
 'sixty three',
 'sixty four',
 'sixty five',
 'sixty six',
 'sixty seven',
 'sixty eight')
```

In [31]:
```
for i in t7:
    print(i)
```

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

In [32]:
```
for i in enumerate(t7):
    print(i)
```

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

In [34]:
```
t6
```

Out[34]:
```
('amit', 20.41, 78, (7+5j), True, 85)
```

In [35]:
```
t6.count(85)
```

Out[35]:
```
1
```

tuple membership

In [36]:
```
t6
```

Out[36]:
```
('amit', 20.41, 78, (7+5j), True, 85)
```

In [37]:
```
'amit' in t6
```

Out[37]:
```
True
```

In [38]:
```
76 in t6
```

Out[38]:
```
False
```

In [39]:
```
85 in t6
```

Out[39]:
```
True
```

In [44]:
```
if 'three' in t6:# Check if 'three' exist in the list
    print('Three is present in the tuple')
else:
    print('Three is not present in the tuple')
```

```
Three is not present in the tuple
```

In [45]:
```
if 'amit' in t6:# Check if 'three' exist 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
```

tuple indexing

In [46]:
```
t6.index(78)
```

Out[46]:
```
2
```

In [47]:
```
t6.index('amit')
```

Out[47]:
```
0
```

In [48]:
```
t6.index(85)
```

Out[48]:
```
5
```

sorting

In [49]:
```
t11=(11,55,44,88,99,33,44,66,22,77,24,34)
t11
```

Out[49]:
```
(11, 55, 44, 88, 99, 33, 44, 66, 22, 77, 24, 34)
```

In [50]:
```
sorted(t11)
```

Out[50]: [11, 22, 24, 33, 34, 44, 44, 55, 66, 77, 88, 99]

In [51]:
```
sorted(t11, reverse=True)
```

Out[51]: [99, 88, 77, 66, 55, 44, 44, 34, 33, 24, 22, 11]

In [52]:
```
sorted(t11, reverse=False)
```

Out[52]: [11, 22, 24, 33, 34, 44, 44, 55, 66, 77, 88, 99]

set

1) Unordered & Unindexed collection of items. 2) Set elements are unique. Duplicate elements are not allowed. 3) Set elements are immutable (cannot be changed). 4) Set itself is mutable. We can add or remove items from it

In [67]:
```
s0=set()
type(s0)
```

Out[67]: set

In [58]:
```
s={4,5,6,7,8} #set of integers
type(s)
```

Out[58]: set

In [59]:
```
s
```

Out[59]: {4, 5, 6, 7, 8}

In [60]:
```
len(s) #length of set
```

Out[60]: 5

In [61]:
```
s1={1,2,3,4,5,4,5,6,1,2} #duplicate are not allowed
s1
```

Out[61]: {1, 2, 3, 4, 5, 6}

In [64]:
```
s2={1.2,1.3,1.4,1.5} #set of float
s2
```

Out[64]:
```
{1.2, 1.3, 1.4, 1.5}
```

In [65]:
```
s3={'amit','vijay','singh','tiwari','jay','pandey'} #set of strings
s3
```

Out[65]:
```
{'amit', 'jay', 'pandey', 'singh', 'tiwari', 'vijay'}
```

In [66]:
```
s4={10,20,'hello',(10,20,30)} #mixed datatypes
s4
```

Out[66]:
```
{(10, 20, 30), 10, 20, 'hello'}
```

In [92]:
```
s5={('one','two','three','four','five','six','seven')}
s5
```

Out[92]:
```
{('one', 'two', 'three', 'four', 'five', 'six', 'seven')}
```

loop through a set

In [93]:
```
for i in s5:
    print(i)
```

```
('one', 'two', 'three', 'four', 'five', 'six', 'seven')
```

In [70]:
```
for i in enumerate(s5):
    print(i)
```

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

set membership

In [94]:
```
s5
```

Out[94]:
```
{('one', 'two', 'three', 'four', 'five', 'six', 'seven')}
```

In [95]:
```
'one' in s5
```

Out[95]:
```
False
```

In [96]:
```
'ten' in s5
```

Out[96]:
```
False
```

In [97]:
```
if 'three' in s5:
    print('three is present in the set')
else:
    print('three is not present in the set')
```

```
three is not present in the set
```

In [98]:
```
if 'eleven' in s5:
    print('eleven is in the set')

else:
    print('eleven is not present in the set')
```

```
eleven is not present in the set
```

Add & Remove functions

In [99]:
```
s5
```

Out[99]:
```
{('one', 'two', 'three', 'four', 'five', 'six', 'seven')}
```

In [83]:
```
s3.add('sumit') #add item to a set using add() functions
```

```
s3
```

Out[83]: {'amit', 'jay', 'pandey', 'singh', 'sumit', 'tiwari', 'vijay'}

In [87]:
```
s3.update(['mohit','rohit','suraj']) # add multiple item using update function
s3
```

Out[87]:
```
{'amit',
 'jay',
 'mohit',
 'pandey',
 'rohit',
 'singh',
 'suraj',
 'tiwari',
 'vijay'}
```

In [88]:
```
s3.remove('suraj') # remove item in a set using remove function
s3
```

Out[88]: {'amit', 'jay', 'mohit', 'pandey', 'rohit', 'singh', 'tiwari', 'vijay'}

In [89]:
```
s3.pop() # remove item in a set using pop function
s3
```

Out[89]: {'amit', 'jay', 'mohit', 'rohit', 'singh', 'tiwari', 'vijay'}

In [90]:
```
s3.discard('tiwari') #remove item in a set using discard function
s3
```

Out[90]: {'amit', 'jay', 'mohit', 'rohit', 'singh', 'vijay'}

In [109…
```
s6=s5.copy()
s6
```

Out[109… {('one', 'two', 'three', 'four', 'five', 'six', 'seven')}

In [110…
```
s6
```

Out[110… {('one', 'two', 'three', 'four', 'five', 'six', 'seven')}

In [113…

```
del s6
s6
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[113], line 2
      1 del s6
----> 2 s6

NameError: name 's6' is not defined
```

In [111…

```
s6
```

Out[111…

```
{('one', 'two', 'three', 'four', 'five', 'six', 'seven')}
```

In [112…

```
s5
```

Out[112…

```
{('one', 'two', 'three', 'four', 'five', 'six', 'seven')}
```

set operations

union

In [114…

```
A={1,2,3,4,5}
B={4,5,6,7,8}
C={8,9,10}
```

In [115…

```
A|B #union of A and B (all elements from both sets no deplicates)
```

Out[115…

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

In [116…

```
A.union(B) #union of a and b
```

Out[116…

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

In [117…
```
A.union(C) #union of a and c
```
Out[117…
```
{1, 2, 3, 4, 5, 8, 9, 10}
```
In [118…
```
A.union(B,C) #union of A,B, and C
```
Out[118…
```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
```
In [119…
```
A.update(B,C)
A
```
Out[119…
```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
```

Intersection

In [120…
```
a={1,2,3,4,5}
b={4,5,6,7,8}
```
In [125…
```
a&b #intersection of a ana b (common items in both sets)
```
Out[125…
```
{4, 5}
```
In [126…
```
a.intersection(b)
```
Out[126…
```
{4, 5}
```
In [127…
```
a.intersection_update(b)
a
```
Out[127…
```
{4, 5}
```

Difference

In [130…
```
a={1,2,3,4,5}
b={4,5,6,7,8}
```

In [131…
```
a-b # set of elements that are only in A but not in B
```

Out[131…
```
{1, 2, 3}
```

In [132…
```
a.difference(b)
```

Out[132…
```
{1, 2, 3}
```

In [140…
```
b-a #set of elements that are only in B but not in A
```

Out[140…
```
{6, 7, 8}
```

In [141…
```
b.difference(a)
```

Out[141…
```
{6, 7, 8}
```

In [142…
```
b.difference_update(a)
b
```

Out[142…
```
{6, 7, 8}
```

Symmetric_difference

In [143…
```
a={1,2,3,4,5}
b={4,5,6,7,8}
```

In [145…
```
a^b #symmetric difference(setof elements in A and B but not in both)
```

Out[145…
```
{1, 2, 3, 6, 7, 8}
```

In [146…
```
a.symmetric_difference(b)
```

Out[146…
```
{1, 2, 3, 6, 7, 8}
```

In [149…
```
a.symmetric_difference_update(b)
a
```

Out[149…
```
{1, 2, 3, 6, 7, 8}
```

subset, superset, disjoint

In [150…
```
a1 = {1,2,3,4,5,6,7,8,9}
b1 = {3,4,5,6,7,8}
c1 = {10,20,30,40}
```

In [154…
```
b1.issubset(a1) # Set B is said to be the subset of set A if all elements of B are
```

Out[154…
```
True
```

In [156…
```
a1.issuperset(b1) # Set A is said to be the superset of set B if all elements of b
```

Out[156…
```
True
```

In [157…
```
c1.isdisjoint(a1)
```

Out[157…
```
True
```

In [158…
```
b1.isdisjoint(a1)
```

Out[158…
```
False
```

other built in function

In [166…
```
a1
```

Out[166…
```
{1, 2, 3, 4, 5, 6, 7, 8, 9}
```

In [167…
```
sum(a1)
```

Out[167… 45

In [168…
```
max(a1)
```

Out[168… 9

In [169…
```
min(a1)
```

Out[169… 1

In [170…
```
len(a1)
```

Out[170… 9

In [164…
```
list(enumerate(a))
```

Out[164… [(0, 1), (1, 2), (2, 3), (3, 7), (4, 6), (5, 8)]

In [171…
```
sorted(a1)
```

Out[171… [1, 2, 3, 4, 5, 6, 7, 8, 9]

In [172…
```
sorted(a1, reverse=True)
```

Out[172… [9, 8, 7, 6, 5, 4, 3, 2, 1]

In [173…
```
sorted(a1, reverse=False)
```

Out[173… [1, 2, 3, 4, 5, 6, 7, 8, 9]

Dictionary

1- Dictionary is a mutable data type in Python. 2- A python dictionary is a collection of key and value pairs separated by a colon (:) & enclosed in curly braces {}. 3- Keys must be unique in a dictionary, duplicate values are allowed.

In [175…
```
d=dict() #Empty dict
d
```

Out[175…
```
{}
```

In [177…
```
d1={}
type(d1)
```

Out[177…
```
dict
```

In [182…
```
mydict_=dict({'a':'one','b':'two','c':'three','d':'four','e':'five','f':'six'})
mydict_
```

Out[182…
```
{'a': 'one', 'b': 'two', 'c': 'three', 'd': 'four', 'e': 'five', 'f': 'six'}
```

In [183…
```
mydict={'a':'one','b':'two','c':'three','d':'four','e':'five','f':'six'}
mydict
```

Out[183…
```
{'a': 'one', 'b': 'two', 'c': 'three', 'd': 'four', 'e': 'five', 'f': 'six'}
```

In [184…
```
mydict.keys()
```

Out[184…
```
dict_keys(['a', 'b', 'c', 'd', 'e', 'f'])
```

In [185…
```
mydict.values()
```

Out[185…
```
dict_values(['one', 'two', 'three', 'four', 'five', 'six'])
```

In [186…
```
mydict.items()
```

Out[186…
```
dict_items([('a', 'one'), ('b', 'two'), ('c', 'three'), ('d', 'four'), ('e', 'five'), ('f', 'six')])
```

In [188…
```
mydict1 = {1:'one' , 2:'two' , 'A':['asif' , 'john' , 'Maria'], 'B':('Bat' , 'cat','hat')}
mydict1
```

Out[188…
```
{1: 'one',
 2: 'two',
 'A': ['asif', 'john', 'Maria'],
 'B': ('Bat', 'cat', 'hat')}
```

In [192…
```
keys={1,2,3,4,5}
value=40
```

```
dict3=dict.fromkeys(keys,value)
dict3
```

Out[192…
```
{1: 40, 2: 40, 3: 40, 4: 40, 5: 40}
```

In [194…
```
key={'a','b','c','d'}
dict2=dict.fromkeys(key)
dict2
```

Out[194…
```
{'d': None, 'b': None, 'c': None, 'a': None}
```

In [196…
```
key={'a','b','c','d'}
value=[10,20]
dict1=dict.fromkeys(key,value)
dict1
```

Out[196…
```
{'d': [10, 20], 'b': [10, 20], 'c': [10, 20], 'a': [10, 20]}
```

In [197…
```
value.append(30)
dict1
```

Out[197…
```
{'d': [10, 20, 30], 'b': [10, 20, 30], 'c': [10, 20, 30], 'a': [10, 20, 30]}
```

Accessing items

In [198…
```
mydict
```

Out[198…
```
{'a': 'one', 'b': 'two', 'c': 'three', 'd': 'four', 'e': 'five', 'f': 'six'}
```

In [200…
```
mydict['d']
```

Out[200…
```
'four'
```

In [202…
```
mydict.get('d')
```

Out[202…
```
'four'
```

In [203…
```
mydict1
```

Out[203…
```
{1: 'one',
 2: 'two',
 'A': ['asif', 'john', 'Maria'],
 'B': ('Bat', 'cat', 'hat')}
```

In [204…
```
mydict1['A']
```

Out[204…
```
['asif', 'john', 'Maria']
```

In [207…
```
mydict1.get(2)
```

Out[207…
```
'two'
```

Add Remove & Change Items

In [208…
```
mydict
```

Out[208…
```
{'a': 'one', 'b': 'two', 'c': 'three', 'd': 'four', 'e': 'five', 'f': 'six'}
```

In [209…
```
mydict['a']='amit'
mydict['b']='singh'
mydict
```

Out[209…
```
{'a': 'amit', 'b': 'singh', 'c': 'three', 'd': 'four', 'e': 'five', 'f': 'six'}
```

In [212…
```
t={'job':'Data Sciense'}
mydict.update(t)
mydict
```

Out[212…
```
{'a': 'amit',
 'b': 'singh',
 'c': 'three',
 'd': 'four',
 'e': 'five',
 'f': 'six',
 'DOB': 1996,
 'job': 'Data Sciense'}
```

In [213…
```
mydict['DOB']=1996 #adding item in the dict
mydict
```

Out[213…
```
{'a': 'amit',
 'b': 'singh',
 'c': 'three',
 'd': 'four',
 'e': 'five',
 'f': 'six',
 'DOB': 1996,
 'job': 'Data Sciense'}
```

In [214…
```
mydict.pop('f')
```

Out[214…
```
'six'
```

In [216…
```
mydict.popitem() # remove random item
```

Out[216…
```
('job', 'Data Sciense')
```

In [217…
```
mydict.popitem()
```

Out[217…
```
('DOB', 1996)
```

In [218…
```
mydict
```

Out[218…
```
{'a': 'amit', 'b': 'singh', 'c': 'three', 'd': 'four', 'e': 'five'}
```

In [219…
```
dict1
```

Out[219…
```
{'d': [10, 20, 30], 'b': [10, 20, 30], 'c': [10, 20, 30], 'a': [10, 20, 30]}
```

In [220…
```
dict1.clear()
```

In [221…
```
dict1
```

Out[221…
```
{}
```

In [223…
```
dict0=dict1.copy()
```

In [224…
```
dict0
```

Out[224…
```
{}
```

In [225…
```
del dict0
```

In [226…
```
dict0
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[226], line 1
----> 1 dict0

NameError: name 'dict0' is not defined
```

In [227…
```
mydict['job']='Data Science'
```

In [228…
```
mydict
```

Out[228…
```
{'a': 'amit',
 'b': 'singh',
 'c': 'three',
 'd': 'four',
 'e': 'five',
 'job': 'Data Science'}
```

In [229…
```
for i in mydict:
    print(i)
```

```
a
b
c
d
e
job
```

In [230…

```python
for i in mydict:
    print(i,':',mydict[i])
```

```
a : amit
b : singh
c : three
d : four
e : five
job : Data Science
```

In [232…

```python
for i in mydict:
    print(mydict[i])
```

```
amit
singh
three
four
five
Data Science
```

Dict Membership

In [233…

```python
mydict
```

Out[233…

```
{'a': 'amit',
 'b': 'singh',
 'c': 'three',
 'd': 'four',
 'e': 'five',
 'job': 'Data Science'}
```

In [235…

```python
'job' in mydict
```

Out[235… True

In [236…
```
'DOB' in mydict
```

Out[236… False

All / Any The all() method returns: True - If all all keys of the dictionary are true False - If any key of the dictionary is false The any() function returns True if any key of the dictionary is True. If not, any() returns False

In [237…
```
all(mydict)
```

Out[237… True

In [238…
```
any(mydict)
```

Out[238… True

In [3]:
```
l=[2,3,1]
l
```

Out[3]: [2, 3, 1]

In [4]:
```
all(l)
```

Out[4]: True

In [5]:
```
any(l)
```

Out[5]: True

In [6]:
```
l.append(0)
```

In [7]:
```
l
```

Out[7]: [2, 3, 1, 0]

In [9]:
```
all(l)
```

Out[9]: False

In [ ]: