24-10-2025

tuple

In [1]:
```
t = ()
t
```

Out[1]: ()

In [2]:
```
type(t)
```

Out[2]: tuple

In [3]:
```
t = (10,20,30)
t
```

Out[3]: (10, 20, 30)

In [4]:
```
t.count(10)
```

Out[4]: 1

In [5]:
```
t.count(20)
```

Out[5]: 1

In [6]:
```
t1 = (10,20,2.2,'ten',True,1+2j)
t1
```

Out[6]: (10, 20, 2.2, 'ten', True, (1+2j))

In [7]:
```
t1.count(20)
```

Out[7]: 1

In [8]:
```
t1 = (10,20,2.2,'ten',True,1+2j,20)
t1
```

Out[8]: (10, 20, 2.2, 'ten', True, (1+2j), 20)

In [9]:
```
t1.count(20)
```

Out[9]: 2

In [10]:
```
t1.index(20)
```

Out[10]: 1

In [11]:
```
print(t)
```

```
print(t1)
```

```
(10, 20, 30)
(10, 20, 2.2, 'ten', True, (1+2j), 20)
```

In [12]:
```
print(len(t))
print(len(t1))
```

```
3
7
```

In [13]:
```
t
```

Out[13]: (10, 20, 30)

In [14]:
```
t[0]
```

Out[14]: 10

In [15]:
```
t[0] = 100
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[15], line 1
----> 1 t[0] = 100

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

In [16]:
```
bank_account = (1234, 'cizp45yi',10000)
bank_account
```

Out[16]: (1234, 'cizp45yi', 10000)

In [17]:
```
bank_account[2] = 200000
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[17], line 1
----> 1 bank_account[2] = 200000

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

In [18]:
```
bank_account.
```

```
  Cell In[18], line 1
    bank_account.
                 ^
SyntaxError: invalid syntax
```

In [19]:
```
t
```

Out[19]: (10, 20, 30)

In [20]:
```
t2 = t * 3
t2
```

Out[20]: (10, 20, 30, 10, 20, 30, 10, 20, 30)

In [21]:
```
t
```

Out[21]: (10, 20, 30)

In [22]:
```
for i in t:
    print(i)
```

```
10
20
30
```

In [ ]:
```

```

SET

In [23]:
```
s = {}
s
```

Out[23]: {}

In [24]:
```
type(s)
```

Out[24]: dict

In [25]:
```
s1 = set()
s1
```

Out[25]: set()

In [26]:
```
s2 = {90, 10, 50, 40, 25, 10, 50}
s2
```

Out[26]: {10, 25, 40, 50, 90}

In [27]:
```
type(s2)
```

Out[27]: set

In [28]:
```
s2
```

Out[28]: {10, 25, 40, 50, 90}

In [29]:
```
s3 = s2.copy()
s3
```

Out[29]: {10, 25, 40, 50, 90}

In [30]:
```
s3
```

Out[30]:
```
{10, 25, 40, 50, 90}
```

In [31]:
```
s3.add(3.4)
```

In [32]:
```
s3
```

Out[32]:
```
{3.4, 10, 25, 40, 50, 90}
```

In [34]:
```
s3.add('nit')
```

In [35]:
```
s3
```

Out[35]:
```
{10, 25, 3.4, 40, 50, 90, 'nit'}
```

In [36]:
```
s3.add(1+2j)
s3.add(True)
```

In [37]:
```
s3
```

Out[37]:
```
{(1+2j), 10, 25, 3.4, 40, 50, 90, True, 'nit'}
```

In [38]:
```
print(s)
print(s1)
print(s2)
print(s3)
```

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

In [39]:
```
s
```

Out[39]:
```
{}
```

In [40]:
```
type(s)
```

Out[40]:
```
dict
```

In [41]:
```
s2
```

Out[41]:
```
{10, 25, 40, 50, 90}
```

In [42]:
```
s3
```

Out[42]:
```
{(1+2j), 10, 25, 3.4, 40, 50, 90, True, 'nit'}
```

In [43]:
```
s3.remove(2000)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[43], line 1
----> 1 s3.remove(2000)

KeyError: 2000
```

In [44]:
```
s3.remove(1+2j)
```

In [45]:
```
s3
```

Out[45]:
```
{10, 25, 3.4, 40, 50, 90, True, 'nit'}
```

In [46]:
```
s3
```

Out[46]:
```
{10, 25, 3.4, 40, 50, 90, True, 'nit'}
```

In [47]:
```
s3.discard(10)
```

In [48]:
```
s3
```

Out[48]:
```
{25, 3.4, 40, 50, 90, True, 'nit'}
```

In [49]:
```
s3.discard(2000)
```

In [50]:
```
s3
```

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

In [51]:
```
s3.pop()
```

Out[51]:
```
True
```

In [52]:
```
s3
```

Out[52]:
```
{25, 3.4, 40, 50, 90, 'nit'}
```

In [53]:
```
s3.pop()
```

Out[53]:
```
3.4
```

In [54]:
```
s3
```

Out[54]:
```
{25, 40, 50, 90, 'nit'}
```

In [55]:
```
s3.pop(0)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[55], line 1
----> 1 s3.pop(0)

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

In [56]:
```
s3[:]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[56], line 1
----> 1 s3[:]

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

In [57]:
```
s3
```

Out[57]: {25, 40, 50, 90, 'nit'}

In [58]:
```
s3[1:]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[58], line 1
----> 1 s3[1:]

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

In [59]:
```
s3
```

Out[59]: {25, 40, 50, 90, 'nit'}

In [60]:
```
s3
```

Out[60]: {25, 40, 50, 90, 'nit'}

In [61]:
```
s3.pop(0)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[61], line 1
----> 1 s3.pop(0)

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

In [62]:
```
s3.pop()
```

Out[62]: 25

In [63]:
```
s3
```

Out[63]: {40, 50, 90, 'nit'}

In [64]:
```
40 in s3
```

Out[64]: True

SET OPERATIONS

In [65]:
```
a = {1,2,3,4,5}
b = {4,5,6,7,8}
c = {8,9,10}
```

In [66]:
```
type(c)
```

Out[66]: set

In [67]:
```
a.union(b)
```

Out[67]: {1, 2, 3, 4, 5, 6, 7, 8}

In [69]:
```
a.union(b, c)
```

Out[69]: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

In [70]:
```
print(a)
print(b)
print(c)
```

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

In [71]:
```
a | b
```

Out[71]: {1, 2, 3, 4, 5, 6, 7, 8}

In [72]:
```
b | c
```

Out[72]: {4, 5, 6, 7, 8, 9, 10}

In [73]:
```
a | b | c
```

Out[73]: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

In [74]:
```
a | c
```

Out[74]: {1, 2, 3, 4, 5, 8, 9, 10}

In [75]:
```
a | c | b
```

Out[75]: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

Intersection

In [76]:
```
a = {1,2,3,4,5}
b = {4,5,6,7,8}
c = {8,9,10}
```

In [77]:
```
a.intersection(b)
```

Out[77]: {4, 5}

In [78]:
```
b.intersection(c)
```

Out[78]: {8}

In [79]:
```
a & b
```

Out[79]: {4, 5}

In [80]:
```
b & c
```

Out[80]: {8}

Difference

In [81]:
```
a = {1,2,3,4,5}
b = {4,5,6,7,8}
c = {8,9,10}
```

In [82]:
```
a.difference(b)
```

Out[82]: {1, 2, 3}

In [83]:
```
b.difference(a)
```

Out[83]: {6, 7, 8}

In [84]:
```
b.difference(c)
```

Out[84]: {4, 5, 6, 7}

In [85]:
```
b - c
```

Out[85]: {4, 5, 6, 7}

In [86]:
```
c - b
```

Out[86]: {9, 10}

In [88]:
```
a - b - c
```

Out[88]: {1, 2, 3}

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:
```
#list
mutable
duplicate is allowed
append(),copy(),insert(),extend(),pop(),
remove the element
list is growable
multiple data type in a list
indexing & slicing is allowed

#Tuple
immutable(unchangeable)
duplication is allowed
remove is not allowed
only 2 function will work(.index,.count)
```