Set

In [10]:
```
s = {}
s
```

Out[10]:
```
{}
```

In [11]:
```
type(s)
```

Out[11]:
```
dict
```

In [12]:
```
s1 = set()
type(s1)
```

Out[12]:
```
set
```

In [13]:
```
s1
```

Out[13]:
```
set()
```

In [14]:
```
s2 = {20, 100, 3, 45}
s2
```

Out[14]:
```
{3, 20, 45, 100}
```

In [15]:
```
s3 = {'z', 'l', 'c', 'e', 'f'}
s3
```

Out[15]:
```
{'c', 'e', 'f', 'l', 'z'}
```

In [16]:
```
s4 = {1, 2.3, 'nit', 1+2j, [1,2,3], (4,5,6), True}
s4
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[16], line 1
----> 1 s4 = {1, 2.3, 'nit', 1+2j, [1,2,3], (4,5,6), True}
      2 s4

TypeError: unhashable type: 'list'
```

In [17]:
```
s5 = {2, 3.4, 'nit', 1+2j, False}
```

In [18]:
```
s5
```

Out[18]:
```
{(1+2j), 2, 3.4, False, 'nit'}
```

In [19]:
```
print(s1)
print(s2)
```

```
print(s3)
print(s5)
```

```
set()
{45, 3, 100, 20}
{'c', 'l', 'f', 'e', 'z'}
{False, 2, 3.4, (1+2j), 'nit'}
```

In [20]:
```
s2.add(30)
```

In [21]:
```
s2
```

Out[21]:
```
{3, 20, 30, 45, 100}
```

In [22]:
```
s2.add(200)
```

In [23]:
```
s2
```

Out[23]:
```
{3, 20, 30, 45, 100, 200}
```

In [24]:
```
s2[1:5]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[24], line 1
----> 1 s2[1:5]

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

In [25]:
```
s5
```

Out[25]:
```
{(1+2j), 2, 3.4, False, 'nit'}
```

In [26]:
```
s4 = s5.copy()
s4
```

Out[26]:
```
{(1+2j), 2, 3.4, False, 'nit'}
```

In [27]:
```
s4
```

Out[27]:
```
{(1+2j), 2, 3.4, False, 'nit'}
```

In [28]:
```
s4.add(2)
```

In [29]:
```
s4
```

Out[29]:
```
{(1+2j), 2, 3.4, False, 'nit'}
```

In [30]:
```
s5
```

Out[30]:
```
{(1+2j), 2, 3.4, False, 'nit'}
```

In [31]:
```
s5.clear()
```

In [32]:
```
s5
```

Out[32]:
```
set()
```

In [33]:
```
del s5
```

In [34]:
```
s4.remove((1+2j))
```

In [35]:
```
s4
```

Out[35]:
```
{2, 3.4, False, 'nit'}
```

In [36]:
```
s4.remove(False, 'nit')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[36], line 1
----> 1 s4.remove(False, 'nit')

TypeError: set.remove() takes exactly one argument (2 given)
```

In [ ]:
```
s3
```

In [ ]:
```
s3.discard('m')   # Discard did not give any error if we did not find any elem
```

In [ ]:
```
s3
```

In [37]:
```
s3.remove('m')  # Got error because m is not family of s3
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[37], line 1
----> 1 s3.remove('m')

KeyError: 'm'
```

In [ ]:
```
s3
```

In [38]:
```
s3.discard('f')    # Here f is eleminated because it is in list
s3
```

Out[38]:
```
{'c', 'e', 'l', 'z'}
```

In [39]:
```
s3
```

Out[39]:
```
{'c', 'e', 'l', 'z'}
```

In [40]:
```
s2
```

Out[40]: {3, 20, 30, 45, 100, 200}

In [41]:
```
s2.pop()
```

Out[41]: 3

In [42]:
```
for i in enumerate(s2):
    print(i)
```

```
(0, 100)
(1, 200)
(2, 45)
(3, 20)
(4, 30)
```

In [43]:
```
s2
```

Out[43]: {20, 30, 45, 100, 200}

In [44]:
```
5 in s2  # Set Membership
```

Out[44]: False

In [45]:
```
45 in s2
```

Out[45]: True

In [46]:
```
s2
```

Out[46]: {20, 30, 45, 100, 200}

In [47]:
```
s3
```

Out[47]: {'c', 'e', 'l', 'z'}

In [48]:
```
s2.update(s3)
```

In [49]:
```
s2
```

Out[49]: {100, 20, 200, 30, 45, 'c', 'e', 'l', 'z'}

SET OPERATIONS

In [50]:
```
s6 = {1,2,3,4,5}
s7 = {4,5,6,7,8}
s8 = {8,9,10}
```

In [51]:
```
s6.union(s7)
```

Out[51]: {1, 2, 3, 4, 5, 6, 7, 8}

In [52]:
```
s6.union(s7, s8)
```

Out[52]: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

In [53]:
```
s6 | s7
```

Out[53]: {1, 2, 3, 4, 5, 6, 7, 8}

In [54]:
```
s6 | s7 | s8
```

Out[54]: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

In [55]:
```
print(s6)
print(s7)
print(s8)
```

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

In [56]:
```
s6.intersection(s7)
```

Out[56]: {4, 5}

In [57]:
```
s6.intersection(s8)
```

Out[57]: set()

In [58]:
```
s7.intersection(s8)
```

Out[58]: {8}

In [59]:
```
s6 & s7
```

Out[59]: {4, 5}

In [60]:
```
print(s6)
print(s7)
print(s8)
```

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

In [61]:
```
s6.difference(s7)
```

Out[61]: {1, 2, 3}

In [62]:
```
s6 - s7
```

Out[62]: {1, 2, 3}

In [63]:
```
s7 - s8
```

Out[63]: {4, 5, 6, 7}

In [64]:
```
print(s6)
print(s7)
print(s8)
```

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

In [65]:
```
s6.symmetric_difference(s7)
```

Out[65]: {1, 2, 3, 6, 7, 8}

In [66]:
```
s10 = {50, 4, 3, 10}
s10
```

Out[66]: {3, 4, 10, 50}

In [67]:
```
print(s10)
```

```
{10, 3, 50, 4}
```

In [68]:
```
print(s10)
```

```
{10, 3, 50, 4}
```

- superset
- subset
- disjoint

In [2]:
```
s11 = {1,2,3,4,5,6,7,8,9}
s12 = {3,4,5,6,7,8}
s13 = {10,20,30,40}
```

In [3]:
```
s12.issubset(s11)
```

Out[3]: True

In [4]:
```
s11.issubset(s12)
```

Out[4]: False

In [5]:
```
s11.issuperset(s12)
```

Out[5]: True

In [6]:
```
s11 = {1,2,3,4,5,6,7,8,9}
s12 = {3,4,5,6,7,8}
s13 = {10,20,30,40}
```

In [7]:
```
s13.isdisjoint(s12)
```

Out[7]: True

In [8]:
```
s13.isdisjoint(s11)
```

Out[8]: True

In [9]:
```
s12 = {1,2,3,4,5}
s13 = {10,20,30}
s14 = {15,25,35}
```

In [10]:
```
s13.issubset(s12)
```

Out[10]: False

In [11]:
```
s12.issuperset(s13)
```

Out[11]: False

In [12]:
```
s14.isdisjoint(s12)
```

Out[12]: True

In [13]:
```
s14.isdisjoint(s13)
```

Out[13]: True

In [14]:
```
s15 = {1,2,3,4,5,6}
s16 = {4,5,6}
s17 = {10,20}
```

In [15]:
```
s16.issubset(s15)
```

Out[15]: True

In [16]:
```
s17.isdisjoint(s15)
```

Out[16]: True

In [17]:
```
s17.isdisjoint(s16)
```

Out[17]: True

In [18]:
```
s15
```

Out[18]: {1, 2, 3, 4, 5, 6}

In [19]:
```
for i in s15:
    print(i)
```

```
1
2
3
4
5
6
```

In [20]:
```
for i in enumerate(s15):
    print(i)
```

```
(0, 1)
(1, 2)
(2, 3)
(3, 4)
(4, 5)
(5, 6)
```

In [21]:
```
s15
```

Out[21]: {1, 2, 3, 4, 5, 6}

In [22]:
```
sum(s15)
```

Out[22]: 21

In [23]:
```
min(s15)
```

Out[23]: 1

Set is completed

Dictionary

In [26]:
```
d = {}
d
```

Out[26]: {}

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

Out[27]: dict

In [28]:
```
d1 = {1 : 'one', 2 : 'two', 3: 'three'}
```

```
d1
```

Out[28]:
```
{1: 'one', 2: 'two', 3: 'three'}
```

In [29]:
```
d1.keys()
```

Out[29]:
```
dict_keys([1, 2, 3])
```

In [30]:
```
d1.values()
```

Out[30]:
```
dict_values(['one', 'two', 'three'])
```

In [31]:
```
d2 = d1.copy()
d2
```

Out[31]:
```
{1: 'one', 2: 'two', 3: 'three'}
```

In [32]:
```
d1.items()
```

Out[32]:
```
dict_items([(1, 'one'), (2, 'two'), (3, 'three')])
```

In [33]:
```
d1[1]
```

Out[33]:
```
'one'
```

In [34]:
```
keys = {'ram' , 'b' , 'c' , 'd'}
value = [10,20,30]
mydict3 = dict.fromkeys(keys , value) # Create a dictionary from a sequence of
mydict3
```

Out[34]:
```
{'c': [10, 20, 30], 'd': [10, 20, 30], 'ram': [10, 20, 30], 'b': [10, 20, 3
0]}
```

In [35]:
```
value.append(50)
mydict3
```

Out[35]:
```
{'c': [10, 20, 30, 50],
 'd': [10, 20, 30, 50],
 'ram': [10, 20, 30, 50],
 'b': [10, 20, 30, 50]}
```

In [36]:
```
range(10)
```

Out[36]:
```
range(0, 10)
```

In [37]:
```
list(range(0,10))
```

Out[37]:
```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

In [38]:
```
list(range(10,20))
```

Out[38]: [10, 11, 12, 13, 14, 15, 16, 17, 18, 19]

In [39]:
```
list(range(10,20,3))
```

Out[39]: [10, 13, 16, 19]

In [40]:
```
list(range(10,20,3,4))
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[40], line 1
----> 1 list(range(10,20,3,4))

TypeError: range expected at most 3 arguments, got 4
```

In [41]:
```
r = range(1,10)
r
```

Out[41]: range(1, 10)

In [42]:
```
for i in r:
    print(i)
```

```
1
2
3
4
5
6
7
8
9
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: