In [1]:
```
s={}
s
```

Out[1]: {}

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

Out[2]: dict

In [3]:
```
s1=set()
s1
```

Out[3]: set()

In [4]:
```
type(s1)
```

Out[4]: set

### add() Functions

In [5]:
```
s1.add(10)
s1.add(20)
s1.add(30)
s1.add(40)
s1
```

Out[5]: {10, 20, 30, 40}

In [7]:
```
s1
```

Out[7]: {10, 20, 30, 40}

In [12]:
```
s1[0]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[12], line 1
----> 1 s1[0]

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

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

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[13], line 1
----> 1 s1[:]

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

In [15]:
```
s2=set()
s2
```

Out[15]: set()

In [16]:
```
s2.add(10)
s2.add(1.2)
s2.add(1+2j)
s2.add(True)
s2.add('nit')
```

In [17]:
```
s2
```

Out[17]: {(1+2j), 1.2, 10, True, 'nit'}

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

```
{40, 10, 20, 30}
{1.2, 'nit', True, (1+2j), 10}
```

**copy() Funtion**

In [20]:
```
s3=s2.copy()
```

In [22]:
```
s3
```

Out[22]: {(1+2j), 1.2, 10, True, 'nit'}

In [23]:
```
s2==s3
```

Out[23]: True

In [24]:
```
s1==s2
```

Out[24]: False

In [25]:
```
print(s1)
print(s2)
print(s3)
```

```
{40, 10, 20, 30}
{1.2, 'nit', True, (1+2j), 10}
{1.2, 'nit', True, (1+2j), 10}
```

**POP() Funtion**

In [30]:
```
s2.pop()
```

Out[30]: (1+2j)

In [34]:
```
s1.pop()    # it directly deleting the first index in set
```

Out[34]: 10

In [28]:
```
s2.pop()
```

Out[28]: True

In [31]:
```
s2
```

Out[31]: {10}

In [33]:
```
s1
```

Out[33]: {10, 20, 30}

### Remove() function

In [35]:
```
s3.remove((1+2j))
```

In [36]:
```
s3
```

Out[36]: {1.2, 10, True, 'nit'}

In [40]:
```
s3.remove(1.2)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[40], line 1
----> 1 s3.remove(1.2)

KeyError: 1.2
```

In [41]:
```
s3
```

Out[41]: {10, True, 'nit'}

### Discard() Function

In [42]:
```
s3.discard(1000)    # it does'nt show error it simply show the original value in
```

In [43]:
```
s3
```

Out[43]: {10, True, 'nit'}

In [44]:
```
s3.discard(True)  # when ever we the give the value to discard it discard the va
```

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

Out[45]: {10, 'nit'}

In [46]:
```
print(s1)
print(s2)
print(s3)
```

```
{20, 30}
{10}
{'nit', 10}
```

### LOOP function in set

In [47]:
```
for i in s1:
    print(i)
```

```
20
30
```

In [48]:
```
for i in s2:
    print(i)
```

```
10
```

In [49]:
```
for i in s3:
    print(i)
```

```
nit
10
```

In [50]:
```
for i in enumerate(s1):
    print(i)
```

```
(0, 20)
(1, 30)
```

In [51]:
```
for i in enumerate(s2):
    print(i)
```

```
(0, 10)
```

In [52]:
```
for i in enumerate(s3):
    print(i)
```

```
(0, 'nit')
(1, 10)
```

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

**Union() Funtion**

In [3]:
```
a.union(b)
```

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

In [4]:
```
a|b
```

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

In [5]:
```
b|c
```

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

In [6]:
```
a|b|c
```

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

In [30]:
```
print(a)
print(b)
print(c)
```

```
{4, 5, 6, 7, 8, 9, 10}
{4, 5, 6, 7, 8}
{8, 9, 10}
```

In [31]:
```
a.update(b,c)
a
```

Out[31]: {4, 5, 6, 7, 8, 9, 10}

### Difference() Funtion

In [32]:
```
a.difference(b)
```

Out[32]: {9, 10}

In [33]:
```
a-b
```

Out[33]: {9, 10}

In [35]:
```
b-c
```

Out[35]: {4, 5, 6, 7}

In [61]:
```
a.difference(c)
```

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

In [62]:
```
b.difference(c)
```

Out[62]: {4, 5, 6, 7}

In [63]:
```
c.difference(b)
```

Out[63]: {9, 10}

### Difference_update() Funtion

In [14]:
```
a.difference_update(c)
a
```

Out[14]: {1, 2, 3, 4, 5, 6, 7}

In [15]:
```
b.difference_update(a)
b
```

Out[15]: {8}

### Intersection

In [16]:
```
a={1,2,3,4,5,6,7}
b={4,5,6,7,8}
```

In [17]:
```
print(a)
print(b)
```

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

In [18]:
```
a&b
```

Out[18]: {4, 5, 6, 7}

In [19]:
```
a.intersection(b)  # In this the intersection we can as two ways 1.intersection
```

Out[19]: {4, 5, 6, 7}

In [21]:
```
b.intersection(a)
```

Out[21]: {4, 5, 6, 7}

In [23]:
```
a.intersection_update(b)
a
```

Out[23]: {4, 5, 6, 7}

In [37]:
```
b.intersection_update(a)
```

### Symmetric Differencre

In [43]:
```
a={1,2,3,4,5,6}
b={4,5,6,7,8}
c={8,7,9,10}
```

In [44]:
```
a.symmetric_difference(b)
```

Out[44]: {1, 2, 3, 7, 8}

In [45]:
```
b.symmetric_difference(a)
```

Out[45]: {1, 2, 3, 7, 8}

In [48]:
```
c.symmetric_difference(b)
```

Out[48]: {4, 5, 6, 9, 10}

In [47]:
```
c.symmetric_difference(a)
```

Out[47]: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

In [50]:
```
a.symmetric_difference_update(b)
a
```

Out[50]: {1, 2, 3, 4, 5, 6}

In [52]:
```
a.symmetric_difference_update(c)
a
```

Out[52]: {1, 2, 3, 4, 5, 6}

In [53]:
```
b.symmetric_difference_update(c)
b
```

Out[53]: {4, 5, 6, 9, 10}

In [55]:
```
c.symmetric_difference_update(a)
c
```

Out[55]: {7, 8, 9, 10}

### Superset() , Subset() and Disjoint()

In [61]:
```
a={1,2,3,4,5,6,7,8,9}
b={3,4,5,6,7,8}
c={10,20,30,40}
```

In [62]:
```
b.issubset(a)
```

Out[62]: True

In [64]:
```
a.issuperset(b)
```

Out[64]: True

In [65]:
```
a.isdisjoint(c)
```

Out[65]: True

In [66]:
```
c.issuperset(b)
```

Out[66]: False

In [68]:
```
b.isdisjoint(c)
```

Out[68]: True

In [ ]:
```

```