set

In [3]:
```
s=set() # empty set
s
```

Out[3]:
```
set()
```

In [6]:
```
s={1,2,3,4,5}
s
```

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

In [7]:
```
len(s)
```

Out[7]:
```
5
```

In [8]:
```
s1={1.1,2.2,3.3,5.5,7.7} # float values
s1
```

Out[8]:
```
{1.1, 2.2, 3.3, 5.5, 7.7}
```

In [9]:
```
s2={1,1,2,2,4,4,2,3,4}  # duplicates are not allowed
s2
```

Out[9]:
```
{1, 2, 3, 4}
```

In [10]:
```
s3={'name','class','subject'} # string values
s3
```

Out[10]:
```
{'class', 'name', 'subject'}
```

In [13]:
```
s4={23,88,42.5,65.77,'name'} # mix data
s4
```

Out[13]:
```
{23, 42.5, 65.77, 88, 'name'}
```

In [14]:
```
s5={'name',33,45,(11,22,33)}
s5
```

Out[14]:
```
{(11, 22, 33), 33, 45, 'name'}
```

In [16]:
```
s6={[11,22,33],'name',33,45} # set does'nt allow mutable items
s6
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[16], line 1
----> 1 s6={[11,22,33],'name',33,45}
      2 s6

TypeError: unhashable type: 'list'
```

In [17]:
```
type(s)
```

Out[17]: set

In [18]:
```
print(type(s))
```

```
<class 'set'>
```

In [19]:
```
myset={'eleven','twelve','thirteen','fourteen','fifteen'}
myset
```

Out[19]: {'eleven', 'fifteen', 'fourteen', 'thirteen', 'twelve'}

loop though a set

In [21]:
```
for i in myset:
    print(i)
```

```
fifteen
thirteen
twelve
eleven
fourteen
```

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

```
(0, 'fifteen')
(1, 'thirteen')
(2, 'twelve')
(3, 'eleven')
(4, 'fourteen')
```

set membership

In [23]:
```
myset
```

Out[23]: {'eleven', 'fifteen', 'fourteen', 'thirteen', 'twelve'}

In [25]:
```
if 'two' in myset:
    print('two is present in myset')
else:
    print('two is not present in myset')
```

```
two is not present in myset
```

In [26]:
```
if 'eleven' in myset:
    print('eleven is present in myset')
else:
    print('eleven is not present in myset')
```

```
eleven is present in myset
```

In [31]:
```
myset
```

Out[31]:
```
{'eleven', 'fifteen', 'fourteen', 'thirteen', 'twelve'}
```

In [32]:
```
'two' in myset
```

Out[32]:
```
False
```

In [49]:
```
'eleven' in myset
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[49], line 1
----> 1 'eleven' in myset

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

Add,remove,discard

In [34]:
```
myset
```

Out[34]:
```
{'eleven', 'fifteen', 'fourteen', 'thirteen', 'twelve'}
```

In [36]:
```
myset.add('two')
myset
```

Out[36]:
```
{'eleven', 'fifteen', 'fourteen', 'thirteen', 'twelve', 'two'}
```

In [37]:
```
myset.add('one','four')  # only one argument should pass
myset
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[37], line 1
----> 1 myset.add('one','four')
      2 myset

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

In [46]:
```
set1={'eleven','twelve','thirteen','fourteen','fifteen'}
set1
```

Out[46]:
```
{'eleven', 'fifteen', 'fourteen', 'thirteen', 'twelve'}
```

In [48]:
```
set1.update(['one','four','six'])  # [multiple values]
set1
```

Out[48]:
```
{'eleven', 'fifteen', 'four', 'fourteen', 'one', 'six', 'thirteen', 'twelve'}
```

In [50]:
```
set1
```

Out[50]:
```
{'eleven', 'fifteen', 'four', 'fourteen', 'one', 'six', 'thirteen', 'twelve'}
```

In [51]:
```
set1.remove('fourteen')
set1
```

Out[51]:
```
{'eleven', 'fifteen', 'four', 'one', 'six', 'thirteen', 'twelve'}
```

In [52]:
```
set1.remove('twelve','fifteen') # only one argument
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[52], line 1
----> 1 set1.remove('twelve','fifteen')

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

In [61]:
```
s2
```

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

In [63]:
```
s2.discard(50)
s2
```

Out[63]:
```
{1, 2, 3, 4}
```

In [64]:
```
s2.discard(4)
s2
```

Out[64]:
```
{1, 2, 3}
```

In [65]:
```
set1
```

Out[65]:
```
{'eleven', 'four', 'one', 'six'}
```

In [67]:
```
set1.clear()
```

In [68]:
```
set1
```

Out[68]:
```
set()
```

In [69]:
```
del set1
```

In [70]:
```
set1
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[70], line 1
----> 1 set1

NameError: name 'set1' is not defined
```

copy set

In [71]:
```
s3
```

Out[71]:
```
{'class', 'name', 'subject'}
```

In [87]:
```
s7=s3
s7
```

Out[87]:
```
{'class', 'name', 'subject'}
```

In [88]:
```
id(s7) ,id(s3)
```

Out[88]:
```
(2092622200416, 2092622200416)
```

In [90]:
```
set2=s3.copy()  # Create a copy of the list
set2
```

Out[90]:
```
{'class', 'name', 'subject'}
```

In [91]:
```
id(set2)
```

Out[91]:
```
2092642843808
```

In [93]:
```
s3.add('roll no')
s3
```

Out[93]:
```
{'class', 'name', 'roll no', 'subject'}
```

In [94]:
```
s7
```

Out[94]:
```
{'class', 'name', 'roll no', 'subject'}
```

In [95]:
```
set2 # id(set2) is diff so if changes made wont be impact
```

Out[95]:
```
{'class', 'name', 'subject'}
```

In [96]:
```
s7.pop()
```

Out[96]:
```
'subject'
```

In [97]:
```
s7
```

Out[97]: {'class', 'name', 'roll no'}

In [99]:
```
s7.pop()
```

Out[99]: 'roll no'

In [100…
```
s7
```

Out[100… {'class', 'name'}

set operators

In [101…
```
a={10,20,30,40,50}
b={50,60,70,80,90}
c={10,20,60}
```

In [102…
```
a.union(b)  # remove the duplicates of a and b gives the value of a,b
```

Out[102… {10, 20, 30, 40, 50, 60, 70, 80, 90}

In [103…
```
b.union(c)
```

Out[103… {10, 20, 50, 60, 70, 80, 90}

In [104…
```
a|b|c
```

Out[104… {10, 20, 30, 40, 50, 60, 70, 80, 90}

In [105…
```
b.update(c)  # update c values in b by removing duplicates
```

In [106…
```
b
```

Out[106… {10, 20, 50, 60, 70, 80, 90}

In [107…
```
print(a)
print(b)
print(c)
```

```
{50, 20, 40, 10, 30}
{80, 50, 10, 20, 70, 90, 60}
{10, 20, 60}
```

In [109…
```
c.update([30,40]) # update 30,40 values in c
c
```

Out[109… {10, 20, 30, 40, 60}

In [110…
```
print(a)
print(b)
print(c)
```

```
{50, 20, 40, 10, 30}
{80, 50, 10, 20, 70, 90, 60}
{40, 10, 20, 60, 30}
```

In [111…
```
a.difference(b) # elements in a but not b
```

Out[111…
```
{30, 40}
```

In [112…
```
b-a # elements in b not in a
```

Out[112…
```
{60, 70, 80, 90}
```

In [113…
```
c.difference(a,b)  # elements in c but not in a and b
```

Out[113…
```
set()
```

In [115…
```
b.difference_update(a)
b
```

Out[115…
```
{60, 70, 80, 90}
```

In [116…
```
a.difference_update(b)
a
```

Out[116…
```
{10, 20, 30, 40, 50}
```

In [118…
```
c.difference_update(b)
c
```

Out[118…
```
{10, 20, 30, 40}
```

In [119…
```
print(a)
print(b)
print(c)
```

```
{50, 20, 40, 10, 30}
{80, 70, 90, 60}
{40, 10, 20, 30}
```

In [120…
```
b.intersection(a) # elements in b same as in a
```

Out[120…
```
set()
```

In [121…
```
c.intersection(a) # elements in c same as in a
```

Out[121…
```
{10, 20, 30, 40}
```

In [123…
```
c.intersection_update(b)
c
```

Out[123…
```
set()
```

In [124…
```
c.intersection_update(a)
c
```

Out[124… set()

In [125…
```
b.intersection_update(a)
b
```

Out[125… set()

In [126…
```
a.intersection_update(b)
a
```

Out[126… set()

In [127…
```
print(a)
print(b)
print(c)
```

```
set()
set()
set()
```

In [128…
```
a.update([1,3,5,7,9])
b.update([10,20,5,7,6])
c.update([5,6,7,8,9])
```

In [130…
```
a.symmetric_difference(b)  # elements in a and b but not in both
```

Out[130… {1, 3, 6, 9, 10, 20}

In [131…
```
b^c
```

Out[131… {8, 9, 10, 20}

In [132…
```
c^a
```

Out[132… {1, 3, 6, 8}

In [133…
```
a.symmetric_difference_update(b)
a
```

Out[133… {1, 3, 6, 9, 10, 20}

In [134…
```
b.symmetric_difference_update(c)
b
```

Out[134… {8, 9, 10, 20}

In [135…
```
c.symmetric_difference_update(a)
c
```

Out[135… {1, 3, 5, 7, 8, 10, 20}

In [136…
```
print(a)
print(b)
print(c)
```

```
{1, 3, 6, 9, 10, 20}
{8, 9, 10, 20}
{1, 3, 5, 7, 8, 10, 20}
```

In [137…
```
b.symmetric_difference_update([1,2])
b
```

Out[137… {1, 2, 8, 9, 10, 20}

In [138…
```
print(a)
print(b)
print(c)
```

```
{1, 3, 6, 9, 10, 20}
{1, 2, 8, 9, 10, 20}
{1, 3, 5, 7, 8, 10, 20}
```

In [139…
```
a.isdisjoint(b) # all elements in a & b should be diff ,True
```

Out[139… False

In [140…
```
a.issubset(b)
```

Out[140… False

In [142…
```
a.issuperset(b)
```

Out[142… False

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

In [144…
```
a.issuperset(b)
```

Out[144… True

In [145…
```
b.issubset(a)
```

Out[145… True

In [146…
```
c.isdisjoint(a)
```

Out[146… True

In [147…
```
a=10,20,30,40,50
```

In [148…
```
sum(a)
```
Out[148… 150

In [149…
```
min(a)
```
Out[149… 10

In [150…
```
max(a)
```
Out[150… 50

In [151…
```
len(a)
```
Out[151… 5

In [152…
```
list(enumerate(a))
```
Out[152… [(0, 10), (1, 20), (2, 30), (3, 40), (4, 50)]

In [155…
```
b=sorted(a,reverse=False)
b
```
Out[155… [10, 20, 30, 40, 50]

In [156…
```
sorted(b)
```
Out[156… [10, 20, 30, 40, 50]

In [157…
```
b=sorted(a,reverse=True)
b
```
Out[157… [50, 40, 30, 20, 10]

Dictionary

In [1]:
```
d={}
d
```
Out[1]: {}

In [2]:
```
type(d)
```
Out[2]: dict

In [3]:
```
print(type(d))
```
```
<class 'dict'>
```

In [4]:
```
d1={1:'one',2:'two',3:'three',4:'four'} # dict with integers
```

In [30]:
```
d2={'a':'one','b':'two','c':'three','d':'four'} # dict with characters
```

In [19]:
```
d3=dict({'one':1,'two':2,'three':3,'four':4}) # using dict()
```

In [6]:
```
d4={1:'one',2:'two',3:['a,b,c'],4:'four',5:{1,2,3}} # mix values
```

In [7]:
```
d4.keys()
```

Out[7]:
```
dict_keys([1, 2, 3, 4, 5])
```

In [8]:
```
d4
```

Out[8]:
```
{1: 'one', 2: 'two', 3: ['a,b,c'], 4: 'four', 5: {1, 2, 3}}
```

In [9]:
```
d4.values()
```

Out[9]:
```
dict_values(['one', 'two', ['a,b,c'], 'four', {1, 2, 3}])
```

In [11]:
```
d4.get(4)
```

Out[11]:
```
'four'
```

In [15]:
```
d4.get(5)
```

Out[15]:
```
{1, 2, 3}
```

In [20]:
```
d3
```

Out[20]:
```
{'one': 1, 'two': 2, 'three': 3, 'four': 4}
```

In [22]:
```
d3.get('three')
```

Out[22]:
```
3
```

In [25]:
```
d3.items()
```

Out[25]:
```
dict_items([('one', 1), ('two', 2), ('three', 3), ('four', 4)])
```

In [31]:
```
d2.items()
```

Out[31]:
```
dict_items([('a', 'one'), ('b', 'two'), ('c', 'three'), ('d', 'four')])
```

In [33]:
```
a={'abc','def','ghi'}
b=dict.fromkeys(a)
b
```

Out[33]:
```
{'def': None, 'ghi': None, 'abc': None}
```

In [34]:
```
a={'abc','def','ghi'}
b={1,2,3}
```

```
c=dict.fromkeys(a,b)
c
```

Out[34]: {'def': {1, 2, 3}, 'ghi': {1, 2, 3}, 'abc': {1, 2, 3}}

In [35]:
```
a={'abc','def','ghi'}
b={'1','2','3'}
c=dict.fromkeys(a,b)
c
```

Out[35]: {'def': {'1', '2', '3'}, 'ghi': {'1', '2', '3'}, 'abc': {'1', '2', '3'}}

copy set

In [40]:
```
c
```

Out[40]: {'def': {'1', '2', '3'}, 'ghi': {'1', '2', '3'}, 'abc': {'1', '2', '3'}}

In [43]:
```
c1=c
c1
```

Out[43]: {'def': {'1', '2', '3'}, 'ghi': {'1', '2', '3'}, 'abc': {'1', '2', '3'}}

In [44]:
```
id(c),id(c1)
```

Out[44]: (2278624630208, 2278624630208)

In [48]:
```
c2=c1.copy()
c2
```

Out[48]: {'def': {'1', '2', '3'}, 'ghi': {'1', '2', '3'}, 'abc': {'1', '2', '3'}}

In [49]:
```
id(c2)
```

Out[49]: 2278625868288

In [50]:
```
print(c)
print(c1)
print(c2)
```

```
{'def': {'1', '2', '3'}, 'ghi': {'1', '2', '3'}, 'abc': {'1', '2', '3'}}
{'def': {'1', '2', '3'}, 'ghi': {'1', '2', '3'}, 'abc': {'1', '2', '3'}}
{'def': {'1', '2', '3'}, 'ghi': {'1', '2', '3'}, 'abc': {'1', '2', '3'}}
```

In [53]:
```
c.pop('abc')
```

Out[53]: {'1', '2', '3'}

In [54]:
```
d3
```

Out[54]: {'one': 1, 'two': 2, 'three': 3, 'four': 4}

In [57]:
```
d3.pop('three')
```

Out[57]:
```
3
```

In [58]:
```
d3.pop('five')
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[58], line 1
----> 1 d3.pop('five')

KeyError: 'five'
```

In [60]:
```
d3
```

Out[60]:
```
{'one': 1, 'two': 2, 'four': 4}
```

In [61]:
```
d3.popitem('four':4)
```

```
  Cell In[61], line 1
    d3.popitem('four':4)
                      ^
SyntaxError: invalid syntax
```

In [62]:
```
d3.clear()
```

In [63]:
```
d3
```

Out[63]:
```
{}
```

In [64]:
```
del d3
```

In [66]:
```
d2
```

Out[66]:
```
{'a': 'one', 'b': 'two', 'c': 'three', 'd': 'four'}
```

In [67]:
```
d2.popitem()  # removes the last item and returns the value
```

Out[67]:
```
('d', 'four')
```

In [68]:
```
d2
```

Out[68]:
```
{'a': 'one', 'b': 'two', 'c': 'three'}
```

In [69]:
```
d2.pop('b') # removes the particular key,value and returns
```

Out[69]:
```
'two'
```

In [70]:
```
d2
```

Out[70]:
```
{'a': 'one', 'c': 'three'}
```

In [72]:
```
d2.popitem()
d2
```

Out[72]:
```
{'a': 'one'}
```

In [76]:
```
d4
```

Out[76]:
```
{1: 'one', 2: 'two', 3: ['a,b,c'], 4: 'four', 5: {1, 2, 3}}
```

In [77]:
```
d4.setdefault(6)
d4
```

Out[77]:
```
{1: 'one', 2: 'two', 3: ['a,b,c'], 4: 'four', 5: {1, 2, 3}, 6: None}
```

In [90]:
```
d4.setdefault(660)
d4
```

Out[90]:
```
{1: 'one', 2: 'two', 3: ['a,b,c'], 4: 'four', 5: {1, 2, 3}, 6: None, 660: None}
```

In [92]:
```
d4.setdefault(6)
d4
```

Out[92]:
```
{1: 'one', 2: 'two', 3: ['a,b,c'], 4: 'four', 5: {1, 2, 3}, 6: None, 660: None}
```

In [95]:
```
d1
```

Out[95]:
```
{1: 'one', 2: 'two', 3: 'three', 4: 'four'}
```

In [110…
```
d1[2]
```

Out[110…
```
'two'
```

In [112…
```
d1[3]='THREE'
d1
```

Out[112…
```
{1: 'one', 2: 'two', 3: 'THREE', 4: 'four'}
```

In [113…
```
d1[5]='five'
d1
```

Out[113…
```
{1: 'one', 2: 'two', 3: 'THREE', 4: 'four', 5: 'five'}
```

In [114…
```
d1[2]='abc'
d1
```

Out[114…
```
{1: 'one', 2: 'abc', 3: 'THREE', 4: 'four', 5: 'five'}
```

In [115…
```
dict1={2:'TWO'}  # updating values in place of abc
d1.update(dict1)
d1
```

Out[115…
```
{1: 'one', 2: 'TWO', 3: 'THREE', 4: 'four', 5: 'five'}
```

In [116…
```
d2
```

Out[116…
```
{'a': 'one'}
```

In [121…
```
d2={'b':'two'} # updating value in place of a
d2.update(d2)
d2
```

Out[121…
```
{'b': 'two'}
```

In [122…
```
d2
```

Out[122…
```
{'b': 'two'}
```

In [123…
```
d2['c']='three' # dding values to dictionary
d2
```

Out[123…
```
{'b': 'two', 'c': 'three'}
```

In [124…
```
d2['a']='one'
d2
```

Out[124…
```
{'b': 'two', 'c': 'three', 'a': 'one'}
```

loop through a dictionary

In [126…
```
d2
```

Out[126…
```
{'b': 'two', 'c': 'three', 'a': 'one'}
```

In [127…
```
for i in d2:
    print(i) # only keys printing
```

```
b
c
a
```

In [131…
```
for i in d2:
    print(i ,':' ,d2[i]) # key value pairing
```

```
b : two
c : three
a : one
```

In [133…
```
for i in d2:
    print(d2[i]) # value printing
```

```
two
three
one
```

In [135…
```
d4
```

Out[135… {1: 'one', 2: 'two', 3: ['a,b,c'], 4: 'four', 5: {1, 2, 3}, 6: None, 660: None}

In [136…
```
for i in d4:
    print(i,':',d4[i])
```

```
1 : one
2 : two
3 : ['a,b,c']
4 : four
5 : {1, 2, 3}
6 : None
660 : None
```

Dictionary membership

In [137…
```
d4
```

Out[137… {1: 'one', 2: 'two', 3: ['a,b,c'], 4: 'four', 5: {1, 2, 3}, 6: None, 660: None}

In [140…
```
1 in d4 # membership is done only for keys
```

Out[140… True

In [139…
```
'one' in d4
```

Out[139… False

In [141…
```
5 in d4
```

Out[141… True

In [142…
```
'two' in d4
```

Out[142… False

All/Any

In [143…
```
d4
```

Out[143… {1: 'one', 2: 'two', 3: ['a,b,c'], 4: 'four', 5: {1, 2, 3}, 6: None, 660: None}

In [144…
```
all(d4) #Will Return false as one value is false (Value 0)
```

Out[144… True

In [145…
```
any(d4)
```

Out[145… True

values

In [146…
```
d4.values()
```

Out[146…
```
dict_values(['one', 'two', ['a,b,c'], 'four', {1, 2, 3}, None, None])
```

In [147…
```
d4.values(6)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[147], line 1
----> 1 d4.values(6)

TypeError: dict.values() takes no arguments (1 given)
```

In [ ]: