SET CREATION

In [2]:
```
s = {1,2,3,4,5}   # SET Members
s
```

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

In [3]:
```
len(s)
```

Out[3]:
```
5
```

In [4]:
```
s_1 = {1,2,3,4,5,5} # Duplicate elements are NOT Allowed
s_1
```

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

In [5]:
```
s1 = {1.79,2.08,99.4,5.45}  #set of FLOAT element
s1
```

Out[5]:
```
{1.79, 2.08, 5.45, 99.4}
```

In [6]:
```
s2 = {'savitri','dev','shankar','prakash sir'}  #set of STRINGS
s2
```

Out[6]:
```
{'dev', 'prakash sir', 'savitri', 'shankar'}
```

In [7]:
```
type(s2)
```

Out[7]:
```
set
```

In [9]:
```
s3 = {10,20,"savi",(11,22,33)}  #MIxed DATA TYPES
s3
```

Out[9]:
```
{(11, 22, 33), 10, 20, 'savi'}
```

In [12]:
```
s3 = {10,20,"savi",[11,22,33]}   # set DOESN'T allow mutable items like LIST
s3
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[12], line 1
----> 1 s3 = {10,20,"savi",[11,22,33]}
      2 s3

TypeError: unhashable type: 'list'
```

In [13]:
```
s4 = set()  #create on EMPTY set
s4
```

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

In [14]:
```
print(type(s4))
```

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

In [15]:
```
type(s4)
```

Out[15]:
```
set
```

In [16]:
```
s_2 = set(('one','two','three','four'))
s_2
```

Out[16]:
```
{'four', 'one', 'three', 'two'}
```

LOOP THROUGH A SET

In [18]:
```
s1 = {'one','two','three','four','five','six','seven','eight'}
for i in s1:
    print(i)
```

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

In [20]:
```
s1 = {1,2,3,4,5,6,7,8}
for i in s1:
    print(i)
```

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

In [21]:
```
for i in s1:
    print(s1)
```

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

In [22]:
```
s1
```

Out[22]: {1, 2, 3, 4, 5, 6, 7, 8}

In [23]:
```
s1 = {'one','two','three','four','five','six','seven','eight'}
s1
```

Out[23]: {'eight', 'five', 'four', 'one', 'seven', 'six', 'three', 'two'}

In [25]:
```
for i in enumerate(s1):
    print(i)
```

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

In [30]:
```

```

SET MEMBERSHIP

In [31]:
```
s1
```

Out[31]: {'eight', 'five', 'four', 'one', 'seven', 'six', 'three', 'two'}

In [32]:
```
'eight' in s1  #check 'eight'  exist in the set or not
```

Out[32]: True

In [33]:
```
'ten' in s1
```

Out[33]: False

In [34]:
```
if 'three' in s1:
    print('three is present in the set')   # check if 'three' exist in the set
else:
    print('three is not present in the set')
```

```
three is present in the set
```

ADD & REMOVE ITEMS

In [35]:
```
s1
```

Out[35]:
```
{'eight', 'five', 'four', 'one', 'seven', 'six', 'three', 'two'}
```

In [37]:
```
s1.add('NINE')
s1
```

Out[37]:
```
{'NINE', 'eight', 'five', 'four', 'one', 'seven', 'six', 'three', 'two'}
```

In [38]:
```
s1.update(['TEN','ELEVEN','TWELVE'])  # add multiple item to a SET using
s1
```

Out[38]:
```
{'ELEVEN',
 'NINE',
 'TEN',
 'TWELVE',
 'eight',
 'five',
 'four',
 'one',
 'seven',
 'six',
 'three',
 'two'}
```

In [39]:
```
s1.remove('NINE')  #REMOVE item in a  SET using REMOVE() method
s1
```

Out[39]:
```
{'ELEVEN',
 'TEN',
 'TWELVE',
 'eight',
 'five',
 'four',
 'one',
 'seven',
 'six',
 'three',
 'two'}
```

In [40]:
```
s1.discard('TEN')  # REMOVE item from a SET using DISCARD() method
s1
```

Out[40]:
```
{'ELEVEN',
 'TWELVE',
 'eight',
 'five',
 'four',
 'one',
 'seven',
 'six',
 'three',
 'two'}
```

In [43]:
```
s1.clear()  # DELETE all items in a SET
```

In [45]:
```
del s1   # DELETE the SET OBJECT
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[45], line 1
----> 1 del s1

NameError: name 's1' is not defined
```

COPY SET

In [48]:
```
s1 = {'one','two','three','four','five','six','seven','eight'}
s1
```

Out[48]:
```
{'eight', 'five', 'four', 'one', 'seven', 'six', 'three', 'two'}
```

In [49]:
```
s = s1  #create a NEW REFERENCE "s"
s
```

Out[49]:
```
{'eight', 'five', 'four', 'one', 'seven', 'six', 'three', 'two'}
```

In [50]:
```
id(s1),id(s)  # the ADDRESS of both s1 & s will be the SAME as
```

Out[50]:
```
(2116059314240, 2116059314240)
```

In [51]:
```
s3 = s1.copy()  # Create a COPY of the list
s3
```

Out[51]:
```
{'eight', 'five', 'four', 'one', 'seven', 'six', 'three', 'two'}
```

In [52]:
```
id(s3)  # the address of s3 will be different from s1 BCZ s3
```

Out[52]:
```
2116059314016
```

In [53]:
```
s1
```

Out[53]:
```
{'eight', 'five', 'four', 'one', 'seven', 'six', 'three', 'two'}
```

In [55]:
```
s1.add('nine')
s1
```

Out[55]:
```
{'eight', 'five', 'four', 'nine', 'one', 'seven', 'six', 'three', 'two'}
```

In [56]:
```
s  # S wil be also impacted
```

Out[56]:
```
{'eight', 'five', 'four', 'nine', 'one', 'seven', 'six', 'three', 'two'}
```

In [57]:
```
s3  # COPY of the set WON'T be imapacte
```

Out[57]: {'eight', 'five', 'four', 'one', 'seven', 'six', 'three', 'two'}

SET OPERATIONS

In [80]:
```
A = {1,2,3,4,5}
B = {4,5,6,7,8}
C = {8,9,10}
```

In [81]:
```
A | B
```

Out[81]: {1, 2, 3, 4, 5, 6, 7, 8}

In [82]:
```
A.union(B)
```

Out[82]: {1, 2, 3, 4, 5, 6, 7, 8}

In [83]:
```
A.union(B,C)
```

Out[83]: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

In [84]:
```
A.update(B,C)
A
```

Out[84]: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

In [86]:
```
A = {1,2,3,4,5}
B = {4,5,6,7,8}
```

In [87]:
```
A & B
```

Out[87]: {4, 5}

In [90]:
```
A.intersection_update(B)
A
```

Out[90]: {4, 5}

In [92]:
```
A = {1,2,3,4,5}
B = {4,5,6,7,8}
```

In [93]:
```
A - B  # SET of elements that are onli in A not B
```

Out[93]: {1, 2, 3}

In [94]:
```
A.difference(B)  # difference of sets
```

Out[94]: {1, 2, 3}

In [95]:
```
B - A  # SET of elements that are onli in A not B
```

Out[95]:
```
{6, 7, 8}
```

In [96]:
```
B.difference(A)
```

Out[96]:
```
{6, 7, 8}
```

In [98]:
```
B.difference_update(A)
B
```

Out[98]:
```
{6, 7, 8}
```

In [100…
```
A ={1,2,3,4,5,}
B = {4,5,6,7,8}
```

In [101…
```
A ^ B  # symmetric difference (set of elements in A and B but not in both ."EX
```

Out[101…
```
{1, 2, 3, 6, 7, 8}
```

In [106…
```
A.symmetric_difference_update(B)
A
```

Out[106…
```
{1, 2, 3, 6, 7, 8}
```

In [107…
```
A.symmetric_difference(B)
A
```

Out[107…
```
{1, 2, 3, 6, 7, 8}
```

SUBSET , SUPERSET & DISJOINT

In [114…
```
A = {1,2,3,4,5,6,7,8,9}
B = {3,4,5,6,7,8}
C = {10,20,30,40}
```

In [115…
```
B.issubset(A)
```

Out[115…
```
True
```

In [116…
```
A.issuperset(A)
```

Out[116…
```
True
```

In [117…
```
C.isdisjoint(A)
```

Out[117…
```
True
```

In [118…
```
B.isdisjoint(A)
```
Out[118… False

In [119…
```
A
```
Out[119… {1, 2, 3, 4, 5, 6, 7, 8, 9}

In [120…
```
sum(A)
```
Out[120… 45

In [121…
```
max(A)
```
Out[121… 9

In [122…
```
min(A)
```
Out[122… 1

In [123…
```
len(A)
```
Out[123… 9

In [124…
```
list(enumerate(A))
```
Out[124… [(0, 1), (1, 2), (2, 3), (3, 4), (4, 5), (5, 6), (6, 7), (7, 8), (8, 9)]

In [126…
```
 D = sorted(A, reverse= True)
D
```
Out[126… [9, 8, 7, 6, 5, 4, 3, 2, 1]

In [128…
```
sorted(A)
```
Out[128… [1, 2, 3, 4, 5, 6, 7, 8, 9]

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: