# Set creation

```
In [1]: s = set() # Empty Set
```

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

```
Out[2]: set
```

```
In [2]: myset = {1,2,3,4,5,6} # Set of integer numbers
        myset
```

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

```
In [4]: len(myset) # Length of the set
```

```
Out[4]: 6
```

```
In [5]: my_set = {1,2,3,1,2,3,2,3,4,5,5}
        my_set   # Duplicate items are not allowed in set
```

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

```
In [6]: myset1 = {1.1,2.2,3.3,4.4,5.5,6.6} # Set of float numbers
        myset1
```

```
Out[6]: {1.1, 2.2, 3.3, 4.4, 5.5, 6.6}
```

```
In [7]: myset2 = {'one', 'two', 'three', 'four', 'five', 'six'} # Set of Strings
        myset2   # it will ordered by alphabate
```

```
Out[7]: {'five', 'four', 'one', 'six', 'three', 'two'}
```

```
In [9]: myset3 = {10,20,'ram',(1,2,3),True,25.5} # Set of Mixed datatypes
        myset3
```

```
Out[9]: {(1, 2, 3), 10, 20, 25.5, True, 'ram'}
```

```
In [10]: myset4 = {10,20,'ram',(1,2,3),True,25.5,[1,2,3]} # Set doesn't allow mutable items
         myseet4
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[10], line 1
----> 1 myset4 = {10,20,'ram',(1,2,3),True,25.5,[1,2,3]} # Set doesn't allow mutable
items like List
      2 myseet4

TypeError: unhashable type: 'list'
```

```
In [11]: myset5 = set((1,2,3,4))
         myset5
```

```
Out[11]: {1, 2, 3, 4}
```

# Loop through a Set

```
In [12]: myset = {1,2,3,4,5,6}

         for i in myset:
             print(i)
```

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

```
In [14]: for i in enumerate(myset):
             print(i)
```

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

# Set Membership

```
In [15]: myset
```

```
Out[15]: {1, 2, 3, 4, 5, 6}
```

```
In [17]: 1 in myset # Check presence of item in set
```

```
Out[17]: True
```

```
In [18]: 7 in myset
```

```
Out[18]: False
```

```
In [19]: if 1 in myset: # Check if 1 is present in set or not
             print("1 is present in myset")
         else:
              print("1 is not present in myset")
```

```
1 is present in myset
```

In [20]:

```
if 7 in myset: # Check if 7 is present in set or not
    print("7 is present in myset")
else:
     print("7 is not present in myset")
```

```
7 is not present in myset
```

# Add & Remove Items

In [21]:

```
myset
```

Out[21]:

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

In [22]:

```
myset.add(7) # Add item to set using add() method
myset
```

Out[22]:

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

In [23]:

```
7 in myset
```

Out[23]:

```
True
```

In [25]:

```
myset.add(8)
myset
```

Out[25]:

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

In [26]:

```
myset.add(9)
myset
```

Out[26]:

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

In [28]:

```
myset.update([10,11,12]) # Add multipe items to a set using update() method
myset
```

Out[28]:

```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12}
```

In [29]:

```
myset.remove(9) # Remove item from set using remove() method
myset
```

Out[29]:

```
{1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12}
```

In [30]:

```
myset.discard(10) # # Remove item from set using discard() method
myset
```

Out[30]:

```
{1, 2, 3, 4, 5, 6, 7, 8, 11, 12}
```

In [32]:

```
myset.discard(13) # discard() method executed without error even item is not presen
myset
```

Out[32]:

```
{1, 2, 3, 4, 5, 6, 7, 8, 11, 12}
```

```
In [33]: myset.clear() # Delete all item in a set
         myset

Out[33]: set()
```

```
In [34]: del myset # Delete the set object
         myset
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[34], line 2
      1 del myset # Delete the set object
----> 2 myset

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

# Copy Set

```
In [35]: myset = {1,2,3,4,5,6,7}
         myset

Out[35]: {1, 2, 3, 4, 5, 6, 7}
```

```
In [36]: myset1 = myset # Create a new reference 'myset1'
         myset1

Out[36]: {1, 2, 3, 4, 5, 6, 7}
```

```
In [37]: id(myset), id(myset1) # The id of bothe set 'myset' & 'myset1' will be same

Out[37]: (1745891980256, 1745891980256)
```

```
In [38]: my_set = myset.copy() # Create a copy of the set
         my_set

Out[38]: {1, 2, 3, 4, 5, 6, 7}
```

```
In [39]: id(my_set), id(myset)

Out[39]: (1745897646432, 1745891980256)
```

```
In [40]: myset.add(9)
         myset

Out[40]: {1, 2, 3, 4, 5, 6, 7, 9}
```

```
In [41]: myset1 # myset1 will be also impacted as it is pointing to the same set

Out[41]: {1, 2, 3, 4, 5, 6, 7, 9}
```

```
In [42]: my_set # Copy of the set won't be impacted due to changes made on the original Set
```

Out[42]:
```
{1, 2, 3, 4, 5, 6, 7}
```

# Set Operations

Union

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

In [44]:
```
a | b # Union of a & b (All elements from bot sets. No Duplicates)
```

Out[44]:
```
{1, 2, 3, 4, 5, 6, 7, 8}
```

In [45]:
```
a.union(b) # Union of a & b
```

Out[45]:
```
{1, 2, 3, 4, 5, 6, 7, 8}
```

In [46]:
```
a | b | c # Union of a , b and c
```

Out[46]:
```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
```

In [47]:
```
a.union(b ,c) # Union of a , b and c
```

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

In [ ]:
```
a.
```

Intersection

In [56]:
```
a = {1,2,3,4,5}
b = {4,5,6,7,8}
c = {8,9,10}
```

In [57]:
```
a & b # Intersection of a and b (Common elements in both sets)
```

Out[57]:
```
{4, 5}
```

In [58]:
```
a.intersection(b)
```

Out[58]:
```
{4, 5}
```

In [59]:
```
a & b & c # Intersection of a, b and c (Common elements in all three sets)
```

Out[59]:
```
set()
```

In [60]:
```
a.intersection(b, c)
```

Out[60]:
```
set()
```

In [73]:

```
a.intersection_update(b)
a
```

Out[73]:

```
set()
```

# Difference

In [62]:

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

In [63]:

```
a - b # set of elements that are only in a but not in b
```

Out[63]:

```
{1, 2, 3}
```

In [64]:

```
a.difference(b)
```

Out[64]:

```
{1, 2, 3}
```

In [66]:

```
a.difference(c)
```

Out[66]:

```
{1, 2, 3, 4, 5}
```

In [67]:

```
b - a
```

Out[67]:

```
{6, 7, 8}
```

In [68]:

```
b.difference(a)
```

Out[68]:

```
{6, 7, 8}
```

In [74]:

```
b.difference_update(a)
b
```

Out[74]:

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

In [75]:

```
A = {1,2,3,4}
B = {3,4,5,6}
A.difference_update(B)
A
```

Out[75]:

```
{1, 2}
```

# Symmetric Difference

In [76]:

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

```
In [77]: a ^ b # Symmetric difference (Set of elements in a and b but not in both.)
```

```
Out[77]: {1, 2, 3, 6, 7, 8}
```

```
In [78]: b.symmetric_difference(a)
```

```
Out[78]: {1, 2, 3, 6, 7, 8}
```

```
In [79]: a.symmetric_difference_update(b)
         a
```

```
Out[79]: {1, 2, 3, 6, 7, 8}
```

# Superset, Subset & Disjoint

```
In [87]: a = {1,2,3,4,5,6,7,8}
         b = {4,5,6,7,8}
         c = {10,20,30}
```

```
In [88]: b.issubset(a) # Set b is said to be the subset of set a if all elements of b are el
```

```
Out[88]: True
```

```
In [89]: a.issuperset(b) # Set a is said to be the superset of set b if all elements of b ar
```

```
Out[89]: True
```

```
In [90]: c.isdisjoint(a) # two sets are said to be disjoint sets if they have no common elem
```

```
Out[90]: True
```

```
In [91]: b.isdisjoint(a)  # two sets are said to be disjoint sets if they have no common ele
```

```
Out[91]: False
```

# Othert Built in functions

```
In [92]: a
```

```
Out[92]: {1, 2, 3, 4, 5, 6, 7, 8}
```

```
In [93]: sum(a)
```

```
Out[93]: 36
```

```
In [94]: max(a)
```

```
Out[94]: 8
```

```
In [95]: min(a)

Out[95]: 1

In [96]: len(a)

Out[96]: 8

In [97]: list(enumerate(a))

Out[97]: [(0, 1), (1, 2), (2, 3), (3, 4), (4, 5), (5, 6), (6, 7), (7, 8)]

In [98]: d = sorted(a, reverse=True)
         d

Out[98]: [8, 7, 6, 5, 4, 3, 2, 1]

In [99]: sorted(d)

Out[99]: [1, 2, 3, 4, 5, 6, 7, 8]

In [ ]:
```