Set

-> A set is an unordered collection of items. Every element is unique (no duplicates).

-> The set itself is mutable. We can add or remove items from it.

Set Creation

In [1]:

```
#set of integers
set_dec = {1, 2, 3}
print(set_dec)

#print type of s
print(type(set_dec))
```

```
{1, 2, 3}
<class 'set'>
```

In [2]:

```
#set doesn't allow duplicates. They store only one instance.
set_duplicates = {1, 2, 3, 1, 4}
print(set_duplicates)
```

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

Set Addtion

The add () method adds an element to the set. If the element already exists, the add () method does not add the element

In [5]:

```
#adding single element to list
set_addition = {1,2,3,4}
set_addition.add(5)

#add multiple elements
set_addition.update([5, 6, 1])
print(set_addition)

#Adding/update set from list and set
set_addition.update([8, 9], {10, 2, 3})
print(set_addition)
```

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

Set Remove

The remove () method takes a single element as an argument and removes it from the set.

In [10]:

```
set_remove = {1, 2, 3, 5, 4}
print(set_remove)

set_remove.remove(4)    #4 is removed from set s

print(set_remove)

#Dicarding element which is not present in set(will get a error)
set_remove.remove(7)
print(set_remove)
```

```
{1, 2, 3, 4, 5}
{1, 2, 3, 5}
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
C:\Users\SHIVAR~1\AppData\Local\Temp/ipykernel_12232/3129843202.py in <module>
      7 
      8 #Dicarding element which is not present in set(will get a error)
----> 9 set_remove.remove(7)
     10 print(set_remove)

KeyError: 7
```

Set Discard

discard () is a built-in method that removes an element from the set only if the item is present in the set.

In [9]:

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

set_discard.discard(2) #2 is dicarded from set s
print(set_discard)

#Dicarding element which is not present in set(will not get error)
set_discard.discard(7)
print(set_discard)
```

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

Python Set Operation

In [11]:

```
set1 = {1, 2, 3, 4, 5}
set2 = {3, 4, 5, 6, 7}

#union of 2 sets using | operator

print(set1 | set2)
```

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

In [12]:

```
#another way of getting union of 2 sets
print(set1.union(set2))
```

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

In [13]:

```
#intersection of 2 sets using & operator
print(set1 & set2)
```

```
{3, 4, 5}
```

In [14]:

```
#use intersection function
print(set1.intersection(set2))
```

```
{3, 4, 5}
```

In [18]:

```
#set Difference: set of elements that are only in set1 but not in set2

print(set1 - set2)

#set Difference: set of elements that are only in set2 but not in set1

print(set2 - set1)
```

```
{1, 2}
{6, 7}
```

In [16]:

```
#use differnce function
print(set1.difference(set2))
```

```
{1, 2}
```

In [17]:

```
"""symmetric difference: set of elements in both set1 and set2
#except those that are common in both."""

#use ^ operator

print(set1^set2)
```

```
{1, 2, 6, 7}
```

In [19]:

```
#find issubset()
x = {"a","b","c","d","e"}
y = {"c","d"}

print("set 'x' is subset of 'y' ?", x.issubset(y)) #check x is subset of y

#check y is subset of x
print("set 'y' is subset of 'x' ?", y.issubset(x))
```

```
set 'x' is subset of 'y' ? False
set 'y' is subset of 'x' ? True
```

In [ ]:

```

```