Sets are like strings but they do not hold Same items

Create a Set

In [7]:
```
s = set()
print(s)

s2 = {526, "myb", 4, "IT"}
print(s2)

#s2 = {} this is not same as set, because its a dictionary

#Sets automatically discard repeated values
s3 = {1,3,5,7,9,5}
print("Repeated Values discarded : ", s3)
```

```
set()
{'myb', 'IT', 4, 526}
Repeated Values discarded :  {1, 3, 5, 7, 9}
```

Set and List

In [9]:
```
#If we want to Eliminate repeated values from list we can convert it to a set
l = [1,2,3,1,4,5,2]
s = set(l)

print(l)
print(s)
```

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

SET OPERATIONS

Union

In [10]:
```
s1 = {1,2,3,5}
s2 = {0,4,6,7,8}

#either
s3 = s1.union(s2)
print(s3)

#OR
s3  = s1 | s2
print(s3)
```

```
{0, 1, 2, 3, 4, 5, 6, 7, 8}
{0, 1, 2, 3, 4, 5, 6, 7, 8}
```

Intersections

In [11]:
```
s1 = {1,2,3,5}
s2 = {1,4,3,7,8}

#either
s3 = s1.intersection(s2)
print(s3)

#OR
s3  = s1 & s2
print(s3)
```

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

Difference

In [13]:
```
s1 = {1,2,3,5}
s2 = {1,4,3,7,8}

#either
s3 = s1.difference(s2)
print(s3)

#OR
s3  = s1 - s2
print(s3)
```

```
{2, 5}
{2, 5}
```

Symteric Difference

(That are in s1 or in s2 but not in both)

In [14]:
```
s1 = {1,2,3,5}
s2 = {1,4,3,7,8}

#either
s3 = s1.symmetric_difference(s2)
print(s3)

#OR
s3  = s1 ^ s2
print(s3)
```

```
{2, 4, 5, 7, 8}
{2, 4, 5, 7, 8}
```

Subset

In [17]:
```
s1 = {1,2,3,5}
s2 = {1,4,3,7,8}

#either
print(s2.issubset(s1))

#OR
print(s2 <= s1)
```

```
False
False
```

Super set

In [18]:
```
s1 = {1,2,3,5}
s2 = {1,4,3,7,8}

print(s2.issuperset(s1))
```

```
False
```

Disjoint

In [20]:
```
s1 = {1,2,3,5}
s2 = {4,7,8}

print(s1.isdisjoint(s2))
```

```
True
```

Create a set and add a single and multiple elements in it

In [22]:
```
s = {1,2,3}

#ADD SINGLE ELEMENT
s.add(4)
print(s)

#ADD MULTIPLE
s.update({5,6,7})
print(s)

s2 = {10,11,0}
s.update(s2)
print(s)
```

```
{1, 2, 3, 4}
{1, 2, 3, 4, 5, 6, 7}
{0, 1, 2, 3, 4, 5, 6, 7, 10, 11}
```

Removing an Element from set

In [31]:

```
s = {1,2,3,5,63}
print("Orignal Set : ",s)

s.remove(1)
print("\nRemoved 1 : ",s)

# IF AN ELEMENT IS NOT IN LIST AND WE CALL REMOVE FOR THAT IT WILL GIVE ERROR
# TO AVOID THIS WE CAN USE DISCARD INSTEAD
s.discard(15)
print("\nDiscarded 15 : ",s)

# POP FUNCTION Removes and returns Fisrt ELEMENT IN SET
print("\npop Element : ",s.pop())
print("\nAfter Pop : ",s)
```

```
Orignal Set :  {1, 2, 3, 5, 63}

Removed 1 :  {2, 3, 5, 63}

Discarded 15 :  {2, 3, 5, 63}

pop Element :  2

After Pop :  {3, 5, 63}
```