In [ ]:
```
# Date 30/12/2025
#here set is also a represent by () but it contradicts with tuple , this is only when you make an
empty tuple
# named as the empty

#Set is of the similar type and not homogeneus type

# set can be represented by()
# obj = (value1, value2) // syntax

# a set has no duplicate records / elements , it will remove the duplicates

s=(1,2,3)
print("The type of s=(1,2,3)",type(s))

# now make it an empty set
s=set()
print("The type of s=set()",type(s))
```

```
The type of s=(1,2,3) <class 'tuple'>
The type of s=set() <class 'set'>
```

In [8]:
```
s=set()
s=set()
for i in range (1,6):
    s.add(i)
print("type of s",type(s))
print("length of s",len(s))
print("min of s",min(s))
print("max of s",max(s))
print("sum of s",sum(s))
print("sort of s",sorted(s))
```

```
type of s <class 'set'>
length of s 5
min of s 1
max of s 5
sum of s 15
sort of s [1, 2, 3, 4, 5]
```

In [10]:
```
# a is a set(1,2) and b = set(3,4) can we concate a and b
a=set()
a.add(1)
a.add(2)

b=set()
b.add(3)
b.add(4)

# print("concated the sets ",(a+b))

a.update(b)
print(a)
```

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

In [11]:
```
# Union / Intersection of sets
a = set()
for i in range (1,6):
    a.add(i)

print("5 in a",5 in a)

print("6 in a",6 in a)

print("6 not in a",6 not in a)
```

```
5 in a True
6 in a False
6 not in a True
```

In [19]:
```
# We can use union() or | (pipe operator)
a = set()
for i in range (1,4):
    a.add(i)

b=set()
b.add(2)
b.add(3)

print("a.union(b)",a.union(b))

print("using the pipe sign ",a|b)

print("Intersection in sets : ",a&b)
```

```
a.union(b) {1, 2, 3}
using the pipe sign  {1, 2, 3}
Intersection in sets :  {2, 3}
```

In [21]:
```
#Difference of the sets:
print("result of a.difference(b) :",a.difference(b))

print("result of a -b : ",(a-b))
```

```
result of a.difference(b) : {1}
result of a -b :  {1}
```

In [31]:
```
#superset is the set that contains all the elements of the subset.
# it has a symbol of >=

#subset is the set that contains just a few elements of the superset

a= set()
a.add(1)
a.add(2)
a.add(3)

b= set()
b.add(1)
b.add(2)

print("a.issuperset(b) (result):",a.issuperset(b))

print("using the symbol a>=b",(a>=b))

print("a.issubset(b) (result):",a.issubset(b))
print("b.issubset(a) (result):",b.issubset(a))

print("using the symbol b<=a",(b<=a))
```

```
a.issuperset(b) (result): True
using the symbol a>=b True
a.issubset(b) (result): False
b.issubset(a) (result): True
using the symbol b<=a True
```

In [ ]:
```
#intersection update
# return a set that has elements that are common as a set
# now both a and b becomes assigned with the intersection value.

print("Early a was :",a)
print("Intersection of a&b",a&b)

a.intersection_update(b)
print("Now a becomes :",a)
```

```
Early a was : {1, 2, 3}
Intersection of a&b {1, 2}
Now a becomes : {1, 2}
```

In [ ]:
```
a = set()
for i in range(1,4):
    a.add(i)

b = set()
for i in range(2,5):
    b.add(i)

print("set a  is =",a)

print("set b  is =",b)

print("The intersection of a and b is ",(a&b))

a.intersection_update(b)

print("Currently done a.intersection_update(b)")

print("now set a  is =",a)

print("now set b  is =",b)
```

```
set a  is = {1, 2, 3}
set b  is = {2, 3, 4}
The intersection of a and b is  {2, 3}
Currently done a.intersection_update(b)
now set a  is = {2, 3}
now set b  is = {2, 3, 4}
```

In [37]:
```
a = set()
for i in range(1,4):
    a.add(i)

b = set()
for i in range(2,5):
    b.add(i)

print("set a  is =",a)

print("set b  is =",b)

print("The difference of a and b is ",(a-b))


a.difference_update(b)

print("Currently done a.difference_update(b)")



print("Currently done a.difference_update(b)")

print("now set a  is =",a)

print("now set b  is =",b)
```

```
set a  is = {1, 2, 3}
set b  is = {2, 3, 4}
The difference of a and b is  {1}
Currently done a.difference_update(b)
Currently done a.difference_update(b)
now set a  is = {1}
now set b  is = {2, 3, 4}
```

In [41]:

```
m = set()
for i in range(1,4):
    m.add(i)
print("set m has :",m)
n = set()
for i in range(2,5):
    n.add(i)
print("set n has :",n)

print("The symmetric difference is : m.symmetric_difference(n) (result)
:\n",m.symmetric_difference(n))
```

```
set m has : {1, 2, 3}
set n has : {2, 3, 4}
The symmetric difference is : m.symmetric_difference(n) (result) :
 {1, 4}
```