set & dict both are defined with flower brackets

In [1]:
```
s ={}
type(s)
```

Out[1]:
```
dict
```

by default it is dictionary

In [2]:
```
s1=set()
s1
```

Out[2]:
```
set()
```

In [3]:
```
s1 = {90 , 4 ,50 ,32 , 3, 1 }
s1
```

Out[3]:
```
{1, 3, 4, 32, 50, 90}
```

In [4]:
```
type(s1)
```

Out[4]:
```
set
```

in python {} -- by default system understands as dict if you create set() then it becomes set

In [5]:
```
s2 = {'z' , 'm' , 'a' , 'd' , 'o'}
```

In [6]:
```
s2
```

Out[6]:
```
{'a', 'd', 'm', 'o', 'z'}
```

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

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

In [8]:
```
print(s1)
print(s2)
```

```
{32, 1, 50, 3, 4, 90}
{'z', 'o', 'a', 'm', 'd'}
```

In [9]:
```
len(s1)
```

Out[9]:
```
6
```

In [10]:
```
len(s2)
```

Out[10]:
```
5
```

In [11]:
```
s3 = {1 , 3.2 , 'nit' , 1+2j , True}
```

In [12]:
```
s3
```

Out[12]:
```
{(1+2j), 1, 3.2, 'nit'}
```

In [13]:
```
s3 = {1 , 3.2 , 'nit' , 1+2j , False}
s3
```

Out[13]:
```
{(1+2j), 1, 3.2, False, 'nit'}
```

In [14]:
```
s1.add(1)
```

In [15]:
```
s1
```

Out[15]:
```
{1, 3, 4, 32, 50, 90}
```

In [16]:
```
s1.add(46)
```

In [17]:
```
s1
```

Out[17]:
```
{1, 3, 4, 32, 46, 50, 90}
```

in sets dublicates are not allowed

In [18]:
```
s1.add(100)
```

In [19]:
```
s1
```

Out[19]:
```
{1, 3, 4, 32, 46, 50, 90, 100}
```

In [20]:
```
s1.add(5)
s1
```

Out[20]:
```
{1, 3, 4, 5, 32, 46, 50, 90, 100}
```

In [21]:
```
print(s1)
```

```
{1, 3, 4, 5, 90, 32, 100, 46, 50}
```

its not manditory that the numbers or elements are in order

In [23]:
```
s3.clear()
```

In [24]:
```
s3
```

Out[24]:
```
set()
```

In [25]:
```
s2
```

Out[25]:
```
{'a', 'd', 'm', 'o', 'z'}
```

In [26]:
```
s4=s1.copy()
s4
```

Out[26]: {1, 3, 4, 5, 32, 46, 50, 90, 100}

In [27]:
```
s1.difference(s2)
```

Out[27]: {1, 3, 4, 5, 32, 46, 50, 90, 100}

In [29]:
```
s3={23,43,53,5}
```

In [30]:
```
s3
```

Out[30]: {5, 23, 43, 53}

s1.difference(s3)

In [31]:
```
s1.difference(s3)
```

Out[31]: {1, 3, 4, 32, 46, 50, 90, 100}

In [32]:
```
s1[1:5]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[32], line 1
----> 1 s1[1:5]

TypeError: 'set' object is not subscriptable
```

In [33]:
```
s1
```

Out[33]: {1, 3, 4, 5, 32, 46, 50, 90, 100}

In [34]:
```
print(s1)
print(s2)
print(s3)
```

```
{1, 3, 4, 5, 90, 32, 100, 46, 50}
{'z', 'o', 'a', 'm', 'd'}
{53, 43, 5, 23}
```

In [35]:
```
s1.pop()
```

Out[35]: 1

In [36]:
```
s1
```

Out[36]: {3, 4, 5, 32, 46, 50, 90, 100}

In [37]:
```
s1.pop()
```

Out[37]: 3

```
In [38]: s1.pop(0)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[38], line 1
----> 1 s1.pop(0)

TypeError: set.pop() takes no arguments (1 given)
```

```
In [39]: s2
```

```
Out[39]: {'a', 'd', 'm', 'o', 'z'}
```

```
In [40]: s1
```

```
Out[40]: {4, 5, 32, 46, 50, 90, 100}
```

```
In [41]: s1.remove(4)
```

```
In [42]: s1
```

```
Out[42]: {5, 32, 46, 50, 90, 100}
```

```
In [43]: s1.remove(46)
```

```
In [44]: s1
```

```
Out[44]: {5, 32, 50, 90, 100}
```

```
In [45]: s1.discard(1000)
```

```
In [46]: s1
```

```
Out[46]: {5, 32, 50, 90, 100}
```

```
In [47]: 3 in s1
```

```
Out[47]: False
```

```
In [48]: 5 in s1
```

```
Out[48]: True
```

```
In [50]: 1000 in s1
```

```
Out[50]: False
```

```
In [51]: s1.discard(5)
```

```
In [52]: s1
```

Out[52]: {32, 50, 90, 100}

1.remove() - remove the element if the element is member , not member it shows error

2.discard() - remove the element if the element is member , not member it does not shows error

In [53]:
```
s1.add(1000)
s1
```

Out[53]: {32, 50, 90, 100, 1000}

set operation

union,intersection,difference,symetric difference,sub set, superset,isdisjoint

In [54]:
```
a={1,2,3,4,5}
b={4,5,6,7,8}
c={8,9,10}
```

In [55]:
```
a.union(b)
```

Out[55]: {1, 2, 3, 4, 5, 6, 7, 8}

In [56]:
```
a.union(b,c)
```

Out[56]: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

In [58]:
```
a | b #union can also be writen like this
```

Out[58]: {1, 2, 3, 4, 5, 6, 7, 8}

In [59]:
```
a | b | c
```

Out[59]: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

intersection

In [61]:
```
print(a)
print(b)
print(c)
```

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

In [62]:
```
a.intersection(b)
```

Out[62]: {4, 5}

In [63]:
```
a.intersection(c)
```

Out[63]: set()

\

if there is no common element set() appears which means empty set

In [65]:
```
a&b #it can also be represented by &
```

Out[65]: {4, 5}

In [66]:
```
print(a)
print(b)
print(c)
```

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

In [67]:
```
a.difference(b)
```

Out[67]: {1, 2, 3}

In [68]:
```
b.difference(a)
```

Out[68]: {6, 7, 8}

In [69]:
```
b - c
```

Out[69]: {4, 5, 6, 7}

In [70]:
```
c-b
```

Out[70]: {9, 10}

In [71]:
```
a-c
```

Out[71]: {1, 2, 3, 4, 5}

In [73]:
```
print(a)
print(b)
print(c)
```

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

In [74]:
```
b.difference_update(c)
```

In [75]:
```
b
```

Out[75]:
```
{4, 5, 6, 7}
```

In [76]:
```
print(a)
print(b)
print(c)
```

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

In [77]:
```
a.symmetric_difference(b)
```

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

In [78]:
```
a^b
```

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

In [ ]: