

Set Data Structure

- ❖ If we want to represent a group of unique values as a single entity then we should go for set.
- ❖ Duplicates are not allowed.
- ❖ Insertion order is not preserved. But we can sort the elements.
- ❖ Indexing and slicing not allowed for the set.
- ❖ Heterogeneous elements are allowed.
- ❖ Set objects are mutable i.e once we create set object we can perform any changes in that object based on our requirement.
- ❖ We can represent set elements within curly braces and with comma separation.
- ❖ We can apply mathematical operations like union, intersection, difference etc on set objects.

We can create set objects by using set() function

```
s=set(any sequence)
```

Important functions of set:

1. add(x):

Adds item x to the set

2. update(x,y,z):

To add multiple items to the set.

Arguments are not individual elements and these are Iterable objects like List, range etc. All elements present in the given Iterable objects will be added to the set.

Q. What is the difference between add() and update() functions in set?

We can use add() to add individual item to the Set, whereas we can use update() function to add multiple items to Set.

add() function can take only one argument whereas update() function can take any number of arguments but all arguments should be iterable objects.

3. copy():

Returns copy of the set.
It is cloned object.

```
s={10,20,30}  
s1=s.copy()  
print(s1)
```

4. pop():

It removes and returns some random element from the set.

5. remove(x):

It removes specified element from the set.
If the specified element not present in the Set then we will get KeyError

6. discard(x):

It removes the specified element from the set.
If the specified element not present in the set then we won't get any error.

7.clear():

To remove all elements from the Set.

Mathematical operations on the Set:

1.union():

$x.union(y)$ ==> We can use this function to return all elements present in both sets

$x.union(y)$ or $x|y$

2. intersection():

$x.intersection(y)$ or $x\&y$

Returns common elements present in both x and y

3. difference():

$x.difference(y)$ or $x-y$
returns the elements present in x but not in y

LET'S PY by shakti Jaiswal

4.symmetric_difference():

`x.symmetric_difference(y)` or `x^y`

Returns elements present in either x or y but not in both

Set Comprehension:

Set comprehension is possible.

```
s={x*x for x in range(5)}  
print(s) #{0, 1, 4, 9, 16}
```

```
s={2**x for x in range(2,10,2)}  
print(s)      #{16, 256, 64, 4}
```

set objects won't support indexing and slicing: