Sets

* Set is an unordered and un indexed collection of multiple elements belonging to different data types.
* They are stored in {}
* We can’t be sure about the order in which they are accessed.

Ex:

sal={12000,9000,45000,34000,23000}

print(sal)

* Sets are un ordered, unchangeable and can’t allow duplicates.
* They can’t be referenced by a key or an index.
* Duplicates if any are ignored.
* len() can be used to get the length of the set.

Using the constructor set() also, we can create sets.

Ex:

sal=set((12000,9000,4000,34000))

print(sal)

o/p: {12000,9000,4000,34000}

Accessing set elements,

for i in sal:

Print(i)

* we can use membership operators “in” and “not in” to check the presence of a member.

Adding items: using add()

sal=set((12000,9000,4000,34000))

sal.add("banana")

print(sal)

o/p:{12000,9000,4000,34000,”banana”}

Concatenate two sets: update()

sal=set((12000,9000,4000,34000))

id={101,102,103}

sal.update(id)

print(sal)

o/p:

{12000,34000,9000,101,102,103,4000}

We can add any iterable (list, tuple,dictionary or a set) to the sets:

Ex: adding dictionary to a set

sal=set((12000,9000,4000,34000))

id={101:'rama',102:'john',103:'sita'} #only keys are added to the set

sal.update(id)

print(sal)

o/p: {12000,34000,9000,101,102,103,4000}

Remove item: remove() or discard()

remove():

sal=set((12000,9000,4000,34000))

id={101:'rama',102:'john',103:'sita'}

sal.update(id)

sal.remove(102)

sal.remove(112) #raises error if element doesn’t exist

print(sal)

discard():

sal=set((12000,9000,4000,34000))

id={101:'rama',102:'john',103:'sita'}

sal.update(id)

sal.discard(102)

sal.discard(112) #no error if element not present also

print(sal)

pop():

removes the last element. But as sets are un ordered, we are not sure about which element gets deleted.

Ex: sal.pop()

clear(): clears the set elements

sal.clear()

del(): deletes the set completely

sal.del()

Sets are unchangeable:

sal=set((12000,9000,4000,34000))

id={101:'rama',102:'john',103:'sita'}

sal.update(id)

sal[2]=2000 #error

print(sal)

Looping thru the set

for i in sal:

Print(i)

Union: returns new set containing elements from both the sets

s1={100,200,300}

s2={1,2,3}

s3=s1.union(s2) #returns new set

print(s3)

o/p:{1,2,3,100,200,300}

update: inserts set2 into s1

s1={100,200,300}

s2={1,2,3}

s1.update(s2)

print(s1)

o/p: {100,200,1,2,3,300}

Note: Both union() and update() excludes any duplicates

Intersection\_update(): keeps only the duplicates present in both the sets

:will not return any set

s1={100,200,300}

s2={1,2,300}

s1.intersection\_update(s2)

print(s1)

o/p: {300}

intersection(): returns a set containing the common elements of both the sets

s1={100,200,300}

s2={1,2,300}

s3=s1.intersection(s2)

print(s3)

o/p:300

symmetric\_difference\_update(): elements that are not present in both the sets

s1={100,200,300}

s2={1,2,300}

s1.symmetric\_difference\_update(s2)

print(s1)

o/p: {1,2,100,200}

symmetric\_difference(): returns the o/p into a set

s1={100,200,300}

s2={1,2,300}

s3=s1.symmetric\_difference(s2)

print(s1)

o/p: {200,100,300}

difference():

s1={100,200,300}

s2={200}

s3=s1.difference(s2)

print(s3)

o/p: {100,300}