**SETS**

1. Sets are used to store multiple items in a single variable
2. Set is an unordered collection of data types(elements) that is iterable.
3. It means the elements may not appear in the same order as they are entered into the set.
4. Sets are unordered so we cannot access its element using index.
5. Set is mutable so we can modify it.
6. No duplicate elements

S={“apple”,”banana”,”apple”}

o/p : s={“apple”,”banana”}

1. Sets are represented using curly brackets { }
2. Example

S={10,20,”priyanka”,30}

**CREATING SET**

A set is created by placing all the items (elements) inside curly braces {}, separated by comma. A set does not accept duplicate elements.

Elements can be of different types except mutable element like list, set or dictionary.

But it can take tuple as an element i.e., set allows immutable elements (it can be string,int,Boolean and tuple)

*s1={10,[2,4,5,6],"priyanka"}*

*s2={10,"priyanka",{10,20}}*

*s3={10,"priyanka",(5,6,3),{1:'a',2:'b'}}*

*s4={10,"priyanka",(5,6,3)}*

*print(s1)*

*print(s2)*

*print(s3)*

*print(s4)*

for unsupported type it gives an error of unhashable type

**CREATING EMPTY SET**

We can create empty set using set()

*a=set()*

*print(type(a))*

using { } it shows dict as a type

*s1={}*

*print(type(s1))*

**ACCESSING and MODIFYING SET ELEMENTS**

Sets are unordered so we cannot access its element using index.

*a = {10, 20,"GeekyShows", "Raj", 40}*

*print(a[0])*

Sets are mutable but as we cannot access elements using index so we cannot modify it

**ADD ITEM**

Once a set is created, you cannot change its items, but you can add new items.

*s1={1,2,3,4}*

*s1.add(7)*

*print(s1)*

**ADD MULTIPLE ITEMS**

**Update()**

To add items from another set into the current set, use the update() method.

The update() method can take tuples,dict, lists, strings or other sets as its argument

*s1={1,2,3,4}*

*s1.update([6,2,1,7])*

*print(s1)*

**REMOVE ITEM**

**Remove()**

To remove an item in a set, use the remove(), or the discard() method.

*s1={1,2,3,4}*

*s1.remove(4)*

*print(s1)*

If the item to remove does not exist, remove() will raise an error.

**Discard()**

*s1={1,2,3,4}*

*s1.discard(5)*

*print(s1)*

 If the item to remove does not exist, discard() will NOT raise an error.

**POP()**

You can also use the pop() method to remove an item, but this method will remove the *last* item. Remember that sets are unordered, so you will not know what item that gets removed.

The return value of the pop() method is the removed item.

*s1={1,2,3,4}*

*s1.pop()*

*print(s1)*

**Clear()**

The clear() method empties the set:

*s1={1,2,3,4}*

*s1.clear()*

*print(s1)*

**Del**

The del keyword will delete the set completel

*s1={1,2,3,4}*

*del s1*

*print(s1)*

**JOIN SETS**

**Union()**

The union() method returns a new set with all items from both sets:

*s1={1,2,3,4}*

*s2={1,2,3,4,5,7,8,10}*

*print(s1.union(s2))*

Both union() and update() will exclude any duplicate items.

**Keep ONLY the Duplicates**

**Intersection()**

The intersection() method will return a *new* set, that only contains the items that are present in both sets.

*s1={1,2,3,4}*

*s2={1,2,3,4,5,7,8,10}*

*print(s1.intersection(s2))*

**intersection\_update()**

The intersection\_update() method will keep only the items that are present in both sets.

*s1={1,2,3,4,11}*

*s2={1,2,3,4,5,7,8,10}*

*print(s1.intersection\_update(s2))*

## Keep All, But NOT the Duplicates

**difference**()

Return a set that contains the items that only exist in set s1, and not in set s2

*s1={"red","purple","pink","green"}*

*s2={"red","blue","purple"}*

*s1.difference(s2)*

*print(s1)*

= {'pink', 'green', 'red', 'purple'}

**symmetric\_difference()**

method will return a new set, that contains only the elements that are NOT present in both sets.

*s1={"red","purple","pink","green"}*

*s2={"red","blue","purple"}*

*s3=s1.symmetric\_difference(s2)*

*print(s3)*

=>{'green', 'blue', 'pink'}

**Symmetric\_differnce\_update()**

Remove the items that are present in both sets, AND insert the items that is not present in both sets

*s1={"red","purple","pink","green"}*

*s2={"red","blue","purple"}*

*s1.symmetric\_difference\_update(s2)*

*print(s1)*

=>{'blue', 'green', 'pink'}

**isdisjoint**()

Returns whether two sets have a intersection or not. Return True if no items in set x is present in set y

*s1={1,2,3,4}*

*s2={1,2,3,4,5}*

*print(s1.isdisjoint(s2))*

**issubset**()

Returns whether another set contains this set or not . Return True if all items in set x are present in set y

*x = {"a", "b", "c"}  
y = {"f", "e", "d", "c", "b", "a"}  
z = x.issubset(y)  
print(z)*

=>True

**issuperset()**

Returns whether this set contains another set or not. Return True if all items set y are present in set x

*x = {"f", "e", "d", "c", "b", "a"}  
y = {"a", "b", "c"}  
z = x.issuperset(y)  
print(z)*

* True