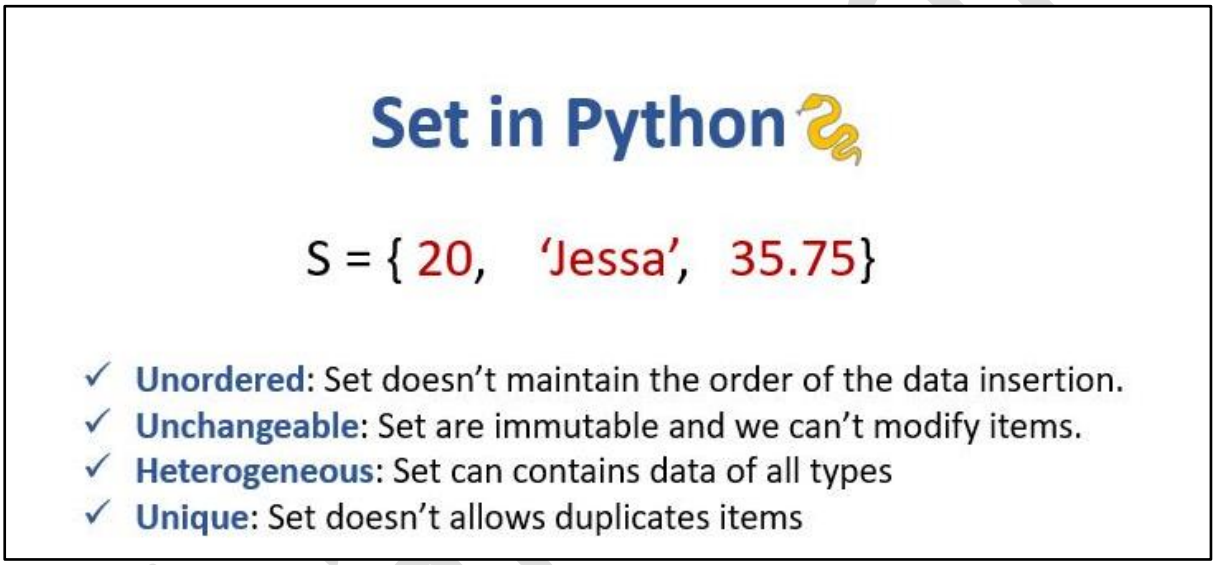
Table of Contents

-Set.....2
Set Creation.....2
#Add Items.....3
#Remove Item.....3
UNION and INTERSECTION.....4
Union:.....4
Intersection.....4
#Set Method:.....4

Chapter 7

Python Collections (Arrays)
There are four collection data types in the Python programming language:
-List is a collection which is ordered and changeable. Allows duplicate members.
-Tuple is a collection which is ordered and unchangeable. Allows duplicate members.
-Set is a collection which is unordered, unchangeable*, and unindexed. No duplicate members.
-Dictionary is a collection which is ordered** and changeable. No duplicate members.

Set



Sets are used to store multiple items in a single variable.
Set is one of 4 built-in data types in Python used to store collections of data, the other 3 are List, Tuple, and Dictionary, all with different qualities and usage.
A set is a collection which is unordered, unchangeable*, and unindexed.
Sets cannot have two items with the same value.
list and dictunaries cannot be added to list because they are changeble

Set Creation

```
mySet = {"Madi", "Perfect" ,2005}
print(type(mySet)) # < class 'set'>
print(mySet)
```

#Add Items

```
# *********Add Items*********
print("*********Add Items*********")
# Empty set is create as:
mySet2 = set()
print(type(mySet2)) # < class 'set'>
mySet2.add("Omar")
mySet2.add("Oily")
mySet2.add(2004)
print(mySet2) # {'Omar', 2004, 'Oily'}
print("Oily" in mySet2) # True
mySet.update(mySet2)
print(mySet) # {'Oily', 2004, 2005, 'Perfect', 'Madi', 'Omar'} random
```

#Remove Item

```
# *********Remove Items*********
# To remove an item in a set, use the remove(), or the discard() method.
print("*********Remove Items*********")
print(mySet) # {'Perfect', 'Oily', 'Madi', 2004, 2005, 'Omar'}
mySet.remove("Oily")
print(mySet) # {'Perfect', 'Madi', 2004, 2005, 'Omar'}
# If the item to remove does not exist, remove() will raise an error.
mySet.discard("Oily")
print(mySet) # {'Perfect', 'Madi', 2004, 2005, 'Omar'}
# If the item to remove does not exist, discard() will NOT raise an error.
x=mySet.pop()
print(x)
# You can also use the pop() method to remove an item, but this method will
remove a random item, so you cannot be sure what item that gets removed.
# The return value of the pop() method is the removed item.

# clear() method empty the set
# del keyword delete the set
```

UNION and INTERSECTION

Union:

The union() method returns a set that contains all items from the original set, and all items from the specified set(s).
You can specify as many sets you want, separated by commas.
It does not have to be a set, it can be any iterable object.
If an item is present in more than one set, the result will contain only one appearance of this item.

```
x = {"a", "b", "c"}
y = {"f", "d", "a"}
z = {"c", "d", "e"}
result = x.union(y, z)
print(result) # {'f', 'e', 'c', 'd', 'b', 'a'}
```

Intersection

The intersection() method returns a set that contains the similarity between two or more sets.
Meaning: The returned set contains only items that exist in both sets, or in all sets if the comparison is done with more than two sets.

```
x = {"a", "b", "c"}
y = {"c", "d", "e"}
z = {"f", "g", "c"}
result = x.intersection(y, z)
print(result) # {'c'}
```

Set Method:

Method	Description
add()	Adds an element to the set
clear()	Removes all the elements from the set
copy()	Returns a copy of the set
difference()	Returns a set containing the difference between two or more sets
difference_update()	Removes the items in this set that are also included in another, specified set
discard()	Remove the specified item
intersection()	Returns a set, that is the intersection of two other sets
intersection_update()	Removes the items in this set that are not present in other, specified set(s)
isdisjoint()	Returns whether two sets have a intersection or not
issubset()	Returns whether another set contains this set or not
issuperset()	Returns whether this set contains another set or not
pop()	Removes an element from the set
remove()	Removes the specified element
symmetric_difference()	Returns a set with the symmetric differences of two sets
symmetric_difference_update()	inserts the symmetric differences from this set and another
union()	Return a set containing the union of sets
update()	Update the set with the union of this set and others