Sets in Python

In [4]:
```
thisset = {}
print(type(thisset))
```
```
<class 'dict'>
```

In [10]:
```
thisset = set({})
print(type(thisset))
```
```
<class 'set'>
```

In [48]:
```
thiset = {"apple","banana","Grapes","guava"}
print(thiset)
```
```
{'banana', 'guava', 'apple', 'Grapes'}
```

In [64]:
```
thiset = {"apple","banana","Grapes","guava"}
thiset
```
Out[64]:
```
{'Grapes', 'apple', 'banana', 'guava'}
```

Duplicates Not Allowed

In [68]:
```
thisset = {1,2,3,4,5,3,2,1}
print(thisset)
```
```
{1, 2, 3, 4, 5}
```

In [76]:
```
thisset = {"apple","banana","guava","sapota","apple"}
print(thisset)
```
```
{'banana', 'guava', 'apple', 'sapota'}
```

The values True and 1 are considered the same and treated as duplicates

In [97]:
```
thisset = {"apples","bananna","True",1,2}
print(thisset)
```
```
{1, 2, 'bananna', 'True', 'apples'}
```

In [99]:
```
thisset = {"apples","bananna",True,1,2}
print(thisset)
```
```
{True, 'apples', 2, 'bananna'}
```

In [103…
```
    thiset = {"A","B","c",0,"false"}
```

```
print(thiset)
```

```
{0, 'B', 'false', 'c', 'A'}
```

In [105…

```
 thiset = {"A","B","c",0,False}
print(thiset)
```

```
{0, 'B', 'c', 'A'}
```

Access Set Items

In [108…

```
thiset = {"apples","banana","cherries"}
print(thiset)
```

```
{'banana', 'apples', 'cherries'}
```

In [116…

```
for i in thiset:
    print(i)
print("banana" in thiset)
```

```
banana
apples
cherries
True
```

Add Set Items

In [121…

```
thiset = {"apples","banana"}
thiset.add("grapes")
print(thiset)
```

```
{'banana', 'apples', 'grapes'}
```

Add Sets

In [124…

```
thiset = {"a","b","c","d"}
set2 = {"e","f","g","h"}
thiset.update(set2)
print(thiset)
```

```
{'b', 'd', 'f', 'e', 'h', 'a', 'g', 'c'}
```

Remove Set Items

by using the remove() method:

In [129…
```
thiset = {"apples","banana","grapes"}
thiset.remove("banana")
print(thiset)
```
```
{'apples', 'grapes'}
```

by using the discard() method:

In [135…
```
thiset = {"apples","banana","grapes"}
thiset.discard("orange")
print(thiset)
```
```
{'banana', 'apples', 'grapes'}
```

pop() method

In [140…
```
thiset = {"a","b","c","d","e"}
thiset.pop()
print(thiset)
```
```
{'d', 'e', 'c', 'a'}
```

In [142…
```
thiset = {"a","b","c","d","e"}
thiset.pop(1)
```
```
---------------------------------------------------------------------------
-
TypeError                                 Traceback (most recent call las
t)
Cell In[142], line 2
      1 thiset = {"a","b","c","d","e"}
----> 2 thiset.pop(1)

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

clear() method e

In [149…
```
thiset = {"a","b"}
thiset.clear()
print(thiset)
```
```
set()
```

del keyword

thiset = {"a","b"} del thiset print(thiset)

Join Sets

union() method

In [158…

```
thiset = {"a","b","c","d"}
set2 = {"e","f","g"}
thiset.union(set2)
```

Out[158…

```
{'a', 'b', 'c', 'd', 'e', 'f', 'g'}
```

Update Method

In [168…

```
thiset = {"a","b","c","d"}
set2 = {"e","f","g"}
thiset.update(set2)
print(thiset)
```

```
{'b', 'd', 'g', 'f', 'e', 'c', 'a'}
```

In [182…

```
set1 = {"a","b","c"}
set2 = {1,2,3}
set1 | set2
```

Out[182…

```
{1, 2, 3, 'a', 'b', 'c'}
```

Join Multiple Sets

In [185…

```
set1 = {"a","b"}
set2 = {"c","d"}
set3 = {"e","f"}
set1.union(set2,set3)
```

Out[185…

```
{'a', 'b', 'c', 'd', 'e', 'f'}
```

In [187…

```
set1 = {"a","b"}
set2 = {"c","d"}
set3 = {"e","f"}
set1 | set2 | set3
```

Out[187…

```
{'a', 'b', 'c', 'd', 'e', 'f'}
```

Join a Set and a Tuple

In [196…
```
set1 = {1,2,3}
tuple1 = ("a","b","c")
lit = ["e","f","g"]
set1.union(tuple1, lit)
```

Out[196…
```
{1, 2, 3, 'a', 'b', 'c', 'e', 'f', 'g'}
```

intersection() method

In [221…
```
set1 = {"a","b","c"}
set2 = {"a","b","c","d","r"}
set1.intersection(set2)
print(set1)
```

```
{'b', 'c', 'a'}
```

In [201…
```
# by using & oprator performaing Intersection
set1 = {1,2,3,}
set2 = {1,2,3,4,5}
set1 & set2
```

Out[201…
```
{1, 2, 3}
```

intersection_update()

In [206…
```
set1 = {"a","b","c"}
set2 = {"a","b","c","d","r"}
set1.intersection_update(set2)
print(set1)
```

```
{'b', 'c', 'a'}
```

In [217…
```
set1 = {"a","b1","c1"}
set2 = {"a","b","c","d","r"}
set1.intersection_update(set2)
print(set1)
```

```
{'a'}
```

In [215…
```
s1 = {1, 2, 3}
s2 = {4, 2, 5}
s1.intersection_update(s2)
print(s1)
```

```
{2}
```

difference() method

In [233…
```
set1 = {1,2,3,4}
set2 = {1,2,3,5,6}
set2.difference(set1)
```
Out[233…
```
{5, 6}
```

In [229…
```
set1 = {1,2,3,4}
set2 = {1,2,3,5,6}
set1 - set2
```
Out[229…
```
{4}
```

In [ ]:
```
```

In [239…
```
set1 = {1,2,3,4}
set2 = {1,2,3,5,6}
set1.difference_update(set2)
print(set1)
```
```
{4}
```

symmetric_difference() method

In [244…
```
set1 = {"apple", "banana", "cherry"}
set2 = {"google", "microsoft", "apple"}
set1.symmetric_difference(set2)
```
Out[244…
```
{'banana', 'cherry', 'google', 'microsoft'}
```

In [246…
```
set1 = {"apple", "banana", "cherry"}
set2 = {"google", "microsoft", "apple"}
set1 ^ set2
```
Out[246…
```
{'banana', 'cherry', 'google', 'microsoft'}
```

sdisjoint() Method

In [249…
```
set1 = {"apple", "banana", "cherry"}
set2 = {"google", "microsoft", "grapes"}
set1.isdisjoint(set2)
```
Out[249…
```
True
```

In [261…
```
set1 = {"apple", "banana", "cherry"}
set2 = {"google", "microsoft", "apple"}
set1.isdisjoint(set2)
```

```
Out[261… False
```

ssubset() Method

```
In [267… set1 = {1,2,3,4}
         set2 = {5,6,7}
         set1.issubset(set2)

Out[267… False

In [277… set1 = {1,2,3}
         set2 = {1,2,3,4,5}
         set1.issubset(set2)

Out[277… True

In [ ]:

In [279… x = {"a", "b", "c"}
         y = {"f", "e", "d", "c", "b"}
         x.issubset(y)

Out[279… False
```

issuperset() Method

```
In [282… x = {"f", "e", "d", "c", "b", "a"}
         y = {"a", "b", "c"}
         x.issuperset(y)

Out[282… True

In [ ]:
```