SETS

Unordered, Unindexed, hence Unchangeable for existing items but new items can be added, Iterable, Duplicate not allowed (wont throw error if duplicate items willbe present while its creation. But will stored as and furthur retrived with only UNIQUE items), can contain any data type items

In [1]:

```
fruits = {"apple", "banana", "cherry"}
fruits.add("orange")
print(fruits)
fruits.add("apple") # doesn't add if elemnt already exixt in set
print(fruits)
```

```
{'apple', 'cherry', 'orange', 'banana'}
{'apple', 'cherry', 'orange', 'banana'}
```

In [2]:

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}
x.update(y) # updates the current set, by adding items from another set (or any other iterable)
print(x)
z=['red', 'green', 'yellow']
x.update(z)
print(x)
```

```
{'cherry', 'microsoft', 'banana', 'google', 'apple'}
{'cherry', 'microsoft', 'banana', 'google', 'green', 'apple', 'red', 'yellow'}
```

In [3]:

```
fruits = {"apple", "banana", "cherry"}
fruits.clear()
print(fruits)

del fruits
try:
    print(fruits)
except:
    print("\'fruits\' set got completely deleted")
```

```
set()
'fruits' set got completely deleted
```

In [4]:

```
fruits = {"apple", "banana", "cherry"}
fruits.remove("banana") # USE discard() INSTEAD
print(fruits)

fruits = {"apple", "banana", "cherry"}
fruits.discard("banana")    # removes the specified item from the set. This method is different from the remove() me
print(fruits)
```

```
{'apple', 'cherry'}
{'apple', 'cherry'}
```

In [5]:

```
fruits = {"apple", "banana", "cherry"}
x = fruits.pop()
print(x)    # removes a random item from the set. This method returns the removed item.
print(fruits)
```

```
apple
{'cherry', 'banana'}
```

In [6]:

```
fruits = {"apple", "banana", "cherry"}
x = fruits.copy()
print(x)
```

```
{'apple', 'cherry', 'banana'}
```

In [7]:

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}
z = x.difference(y) # returns set that contains items that exist only in the first set, and not in both sets.
print(z)

x = {"apple", "banana", "cherry"}
y = {"apple", "banana", "cherry"}
z = y.difference(x)
print(z)
```

```
{'cherry', 'banana'}
set()
```

In [8]:

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}
x.difference_update(y)  # The difference_update() method is different from the difference() method, because the diff
print(x)
print(y)
```

```
{'cherry', 'banana'}
{'apple', 'microsoft', 'google'}
```

In [9]:

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}
z = x.union(y)  # The union() method returns a set that contains all items from the original set, and all items from
print(z)

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

```
{'cherry', 'microsoft', 'banana', 'google', 'apple'}
{'a', 'd', 'c', 'e', 'b', 'f'}
```

In [10]:

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}
z = x.intersection(y)   # set.intersection(set1[, set2 ... etc]) . returns a set that contains the similarity betwee
print(z)


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

```
{'apple'}
{'c'}
```

In [11]:

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}
x.intersection_update(y)    # set.intersection_update(set1, set2 ... etc) . Removes the items that is not present in
print(x)

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

```
{'apple'}
{'c'}
```

In [12]:

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}
z = x.symmetric_difference(y)
print(z)

# returns a set that contains all items from both set, but not the items that are present in both sets, except items
```

```
{'cherry', 'google', 'microsoft', 'banana'}
```

In [13]:

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}
x.symmetric_difference_update(y)    # updates the original set by removing items that are present in both sets, and
print(x)
```

```
{'cherry', 'google', 'microsoft', 'banana'}
```

In [14]:

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "facebook"}
z = x.isdisjoint(y) # # returns True if none of the items are present in both sets, otherwise it returns False.
print(z)

x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}
z = x.isdisjoint(y)
print(z)

x = set()
y = set()
z = x.isdisjoint(y)
print(z)
```

```
True
False
True
```

In [15]:

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

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

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

```
True
False
True
```

In [16]:

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

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

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

```
True
False
False
```