python

Set Methods

Sets are unordered & mutable collections of unique elements

add

Add an element to Set. Will not add if element already exists

```
> x = {'Jack', 'Csaba', 'Nat'}
> x.add('Annie')
> x
{'Csaba', 'Annie', 'Jack', 'Nat'}
```

clear

Remove all elements in set

```
> x = {'Emma', 'Mike'}
> x.clear()
> x
set()
```

copy

Return copy of set

```
> x = {'Layla', 'Roxanna'}
> y = x.copy()
> y
{'Layla', 'Roxanna'}
```

difference

Return set with difference between two sets. i.e. exist only in the first set & not in both sets

```
> x = {'Tabs', 'Oliver', 'Coner'}
> y = {'Sam', 'Oliver', 'Janine'}
> z = x.difference(y)
> z
{'Coner', 'Tabs'}
```

difference_update

Same as difference but removes unwanted items instead of returning new set

```
> x = {'Tabs', 'Oliver', 'Coner'}
> y = {'Sam', 'Oliver', 'Janine'}
> x.difference_update(y))
> x
{'Coner', 'Tabs'}
```

discard

Remove item from set. Will not throw error if item doesn't exist

```
> x = {'John', 'Aaron', 'Luise'}
> x.discard('John')
> x
{'Aaron', 'Luise'}
```

intersection

Return set containing only items that exist in all sets. Can pass multiple parameters

```
> x = {'Rosie', 'Pratham', 'Sean'}
> y = {'Pratham', 'Yuri', 'Vitto'}
> z = x.intersection(y)
> z
{'Pratham'}
```

intersection_update

Same as intersection but removes unwanted items instead of returning new set

```
> x = {'Rosie', 'Pratham', 'Sean'}
> y = {'Pratham', 'Yuri', 'Vitto'}
> x.intersection_update(y)
> x
{'Pratham'}
```

isdisjoint

Return True if none of the items are in both sets

```
> x = {'Kolade', 'Kevin', 'Brad'}
> y = {'Sarah', 'Jon', 'Favor'}
> z = x.isdisjoint(y)
> z
True
```

issubset

Return True if all items in original set exist in specified set

```
> x = {'Meet', 'Khurram'}
> y = {'Dan', 'Khurram', 'Meet'}
> z = x.issubset(y)
> z
True
```

issuperset

Return True if all items in specified set exist in original set

```
> x = {'Ali', 'Sheldon', 'Rodrigo'}
> y = {'Sheldon', 'Rodrigo'}
> z = x.issuperset(y)
> z
True
```

pop

Remove and return random item from set

```
> x = {'Akshaya', 'Irina', 'Alex'}
> y = x.pop()
> print(x, y)
{'Alex', 'Akshaya'} Irina
```

remove

Same as discard but will raise error if item does not exist

```
> x = {'Aaron', 'John', 'Luise'}
> x.remove('John')
> x
{'Aaron', 'Luise'}
```

symmetric_difference

Return set containing all items from both sets, but not those present in both sets

```
> x = {'Meena', 'Sam', 'Kaarthik'}
> y = {'Simon', 'Sam', 'Meena'}
> z = x.symmetric_difference(y)
> z
{'Kaarthik', 'Simon'}
```

symmetric_difference_update

Same as symmetric_difference but updates the original set rather than returning new one

```
> x = {'Meena', 'Sam', 'Kaarthik'}
> y = {'Simon', 'Sam', 'Meena'}
> x.symmetric_difference_update(y)
> x
{'Kaarthik', 'Simon'}
```

union

Return set containing items from original set & items from specified set(s) or iterable(s)

```
> x = {'Debbie', 'Jess','Anna'}
> y = {'Kola', 'Anna'}
> z = x.union(y)
> z
{'Anna', 'Kola', 'Jess', 'Debbie'}
```

update

Update original set by adding items from another set or iterable

```
> x = {'Alexandria', 'David'}
> y = {'Bridget', 'David'}
> x.update(y)
> x
{'Bridget', 'David', 'Alexandria'}
```

Reference

- W3Schools.com

By @AbzAaron

Feel Free to Buy Me A Coffee