Set Methods

Table of contents

01

Adding

Adding

01 — `update()` — Adds **multiple elements** from an iterable

02 — `add()` — Adds a **single element** to the set.

Adding – update()

Python	Output

`s = {1, 2, 3}`
`s.update([4, 5, 6])`
`print(s)`
	`{1, 2, 3, 4, 5, 6}`

update() – Adds multiple elements

Adding – add()

Python	Output

`s = {1, 2, 3}`
`s.add(4)`
`print(s)`
	`{1, 2, 3, 4}`

add() – Adds a single elements

02

Removing Elements

Removing Elements

01 — `discard()` — Removes a **specific** element **without** raising an **error** if it's not found.

02 — `remove()` — Removes a **specific** element, **raises KeyError** if not found.

03 — `pop()` — Removes and returns a **random element**, raises **KeyError** if set is empty.

04 — `clear()` — Removes **all elements** from the set.

Removing Elements – discard()

Python	Output

`s = {1, 2, 3}`
`s.discard(2)`
`s.discard(5)`
`print(s)`
	`{1, 3}`
	
	`# No error if 5 is not found`

discard() – Removes an element (no error if not found)

Removing Elements – remove()

Python	Output

`s = {1, 2, 3}`
`s.remove(2)`
`s.remove(5)`
`print(s)`
	`{1, 3}`
	
	`# KeyError: 5 not found`

remove() – Removes an element (raises error if not found)

Removing Elements – pop()

Python	Output

`s = {1, 2, 3, 4}`
`print(s.pop())`
 	
`print(s)`
	`# Removes and prints a random element`
	
	`# Remaining elements`

pop() – Removes and returns a random element

Removing Elements – clear()

Python	Output

`s = {1, 2, 3}`
`s.clear()`
`print(s)`
	`set()`

clear() – Removes all elements from the set

03

Set Operations

Set Operations

01 — `union()` — Returns a set containing **all** elements from **both** sets (no duplicates).

02 — `intersection()` — Returns a set containing elements **common** in **both** sets.

03 — `difference()` — Returns elements that are **in the first** set but **not in the second**.

04 — `symmetric_difference()` — Returns elements that are **in either** set but **not both**.

Set Operations

05 — `intersection_update()` — Keeps **only elements** found in both sets

06 — `difference_update()` — Removes elements found in **another** set

07 — `symmetric_difference_update` — Keeps only **non-common** elements

Set Operations – union()

Python	Output

`s1 = {1, 2, 3}`
`s2 = {3, 4, 5}`
`print(s1.union(s2))`
	`{1, 2, 3, 4, 5}`

union() – Returns all unique elements from both sets

Set Operations – intersection()

Python	Output

`s1 = {1, 2, 3}`
`s2 = {2, 3, 4}`
`print(s1.intersection(s2))`
	`{2, 3}`

intersection() – Returns elements common in both sets

Set Operations – difference()

Python	Output

`s1 = {1, 2, 3, 4}`
`s2 = {3, 4, 5, 6}`
`print(s1.difference(s2))`
	`{1, 2}`

difference() – Returns elements in s1 but not in s2

Set Operations - symmetric_difference()

Python	Output

`s1 = {1, 2, 3}`
`s2 = {2, 3, 4}`
`print(s1.symmetric_difference(s2))`
	`{1, 4}`

symmetric_difference() – Returns elements that are in either set but not both.

Set Operations – intersection_update()

Python	Output

`s1 = {1, 2, 3}`
`s2 = {2, 3, 4}`
`s1.intersection_update(s2)`
`print(s1)`
	`{2, 3}`

intersection_update() – Keeps only elements found in both sets.

Set Operations - difference_update()

Python	Output

`s1 = {1, 2, 3, 4}`
`s2 = {3, 4, 5}`
`s1.difference_update(s2)`
`print(s1)`
	`{1, 2}`

difference_update() – Removes elements found in another set.

Set Operations – symmetric_difference_update()

Python	Output

`s1 = {1, 2, 3}`
`s2 = {2, 3, 4}`
`s1.symmetric_difference_update(s2)`
`print(s1)`
	`{1, 4}`

symmetric_difference_update() – Keeps only non-common elements

04

Relationship

Relationship

01 — `issubset()` — Checks if the set is a **subset** of another.

02 — `issuperset()` — Checks if the set is a **superset** of another.

03 — `isdisjoint()` — Checks if two sets have **no common elements**.

Relationship – issubset()

Python	Output

`s1 = {1, 2}`
`s2 = {1, 2, 3, 4}`
`print(s1.issubset(s2))`
	`True`

issubset() – Checks if all elements of **s1** are in **s2**

Relationship - issuperset()

Python	Output

`s1 = {1, 2, 3, 4}`
`s2 = {2, 3}`
`print(s1.issuperset(s2))`
	`True`

issuperset() – Checks if **s1** contains all elements of **s2**

Relationship - isdisjoint()

Python	Output

`s1 = {1, 2, 3}`
`s2 = {4, 5, 6}`
`print(s1.isdisjoint(s2))`
	`True`

isdisjoint() – Checks if **s1** and **s2** have no common elements

05

Utility

Utility

01 — `copy()` — Returns a **shallow copy** of the set.

Utility – copy()

Python	Output

`s1 = {1, 2, 3}`
`s2 = s1.copy()`
`print(s2)`
	`{1, 2, 3}`

copy() – Returns a **shallow copy** of the set

Summary Table

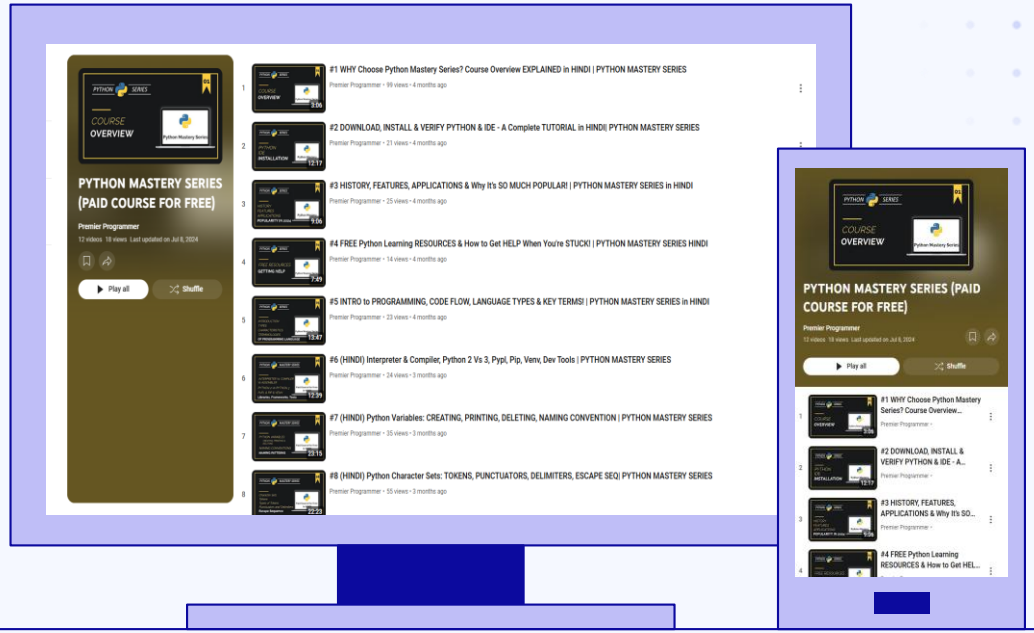
Category	Methods			
Adding	add()		update()	
Removing	discard()		remove()	
	pop()		clear()	
Set Operations	union()	intersection()	difference()	symmetric_difference()
	intersection_update()		difference_update()	symmetric_difference_update()
Relationship	Issubset()		Issuperset()	Isdisjoint()
Utility	Copy()			

Practice Set - 4

Download Link in **Description** and **Pinned Comment**

WATCH

Level up your coding with each episode in this focused Python series.



Next Video!

Practice Set – 4
Solution